RANKING, LABELING, AND SUMMARIZING SHORT TEXT IN SOCIAL MEDIA

A Dissertation

by

ELHAM KHABIRI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	James Caverlee
Committee Members,	Frank Shipman
	Ricardo Gutierrez Osuna
	Patrick Burkart
Department Head,	Duncan Walker

May 2013

Major Subject: Computer Science

ABSTRACT

One of the key features driving the growth and success of the Social Web is large-scale participation through user-contributed content – often through *short text in social media*. Unlike traditional long-form documents – e.g., Web pages, blog posts – these short text resources are typically quite brief (on the order of 100s of characters), often of a personal nature (reflecting opinions and reactions of users), and being generated at an explosive rate. Coupled with this explosion of short text in social media is the need for new methods to organize, monitor, and distill relevant information from these large-scale social systems, even in the face of the inherent "messiness" of short text, considering the wide variability in quality, style, and substance of short text generated by a legion of Social Web participants.

Hence, this dissertation seeks to develop new algorithms and methods to ensure the continued growth of the Social Web by enhancing how users engage with short text in social media. Concretely, this dissertation takes a three-fold approach:

- First, this dissertation develops a learning-based algorithm to automatically *rank* short text comments associated with a Social Web object (e.g., Web document, image, video) based on the expressed preferences of the community itself, so that low-quality short text may be filtered and user attention may be focused on highly-ranked short text.

- Second, this dissertation organizes short text through *labeling*, via a graph-based framework for automatically assigning relevant labels to short text. In this way meaningful semantic descriptors may be assigned to short text for improved classification, browsing, and visualization.

- Third, this dissertation presents a cluster-based *summarization* approach for extracting high-quality viewpoints expressed in a collection of short text, while maintaining diverse viewpoints. By summarizing short text, user attention may quickly assess the aggregate viewpoints expressed in a collection of short text, without the need to scan each of possibly thousands of short text items.

DEDICATION

To My Mom, Dad, and Roozbeh.

ACKNOWLEDGEMENTS

I would like to express my thanks and gratitude to all who supported and helped me through writing this dissertation.

I am deeply indebted to my advisor, Dr. James Caverlee for his unlimited help and support throughout my research; After any single meeting with him I felt more motivated and more eager about my research. He taught me first how to be a better person and second how to do a better research. His motivation and suggestions were always with me through the completion of my PhD. Without his continuous help and support this work would not have been possible.

Many thanks to my lab-mates for their support and much of helpful discussions. I would like to thank Zhiyuan Cheng for his patience and constructive suggestions. Many thanks to Chiao-Fang Hsu and Krishna Kamath for wonderful discussions we had during our collaborations. Many thanks to Yuan Liang, Kyumin Lee, Jeff McGee for making our lab a joyful place to live. I should also thank Jeremy Kelley. His presence in our lab was a gift for all of us.

I wish to thank my parents for their love, support and confidence throughout the thirty years. I owe them much of what I have become now. I would like to thank my Mom for all her continuous prayers and insightful advice and my Dad for his encouraging words which have always given me hope and courage.

And finally my patient and loving husband, Roozbeh, who has been always there for me. He taught me how to become and stay passionate about my research. This work was impossible without his love, understanding, care and support.

I dedicate this work to my parents and my husband, to gratitude their love, patience and support during these years.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

The Social Web is one of the early successes in the emerging social computing paradigm. Prominent Social Web examples include large-scale information sharing communities (e.g., Wikipedia), social media sites (e.g., YouTube), and web-based social networks (e.g., Facebook), each centered around user-contributed content and community-based information sharing.

One of the key features driving the growth and success of the Social Web is large-scale participation through user-contributed content – often through *short text in social media*. Unlike traditional long-form documents – e.g., Web pages, blog posts – these short text resources are typically quite brief (on the order of 100s of characters), often of a personal nature (reflecting opinions and reactions of users), and are being generated at an explosive rate. As one example, Twitter has rapidly grown from handling 5,000 tweets per day in 2007 to 50 million tweets per day in 2010 to 140 million tweets per day in 2011. During the recent run-up and immediate aftermath of President Obama's announcement about Osama Bin Laden, Twitter boasted a peak of 5,000 tweets per second (corresponding to 432 million tweets per day) and a sustained average rate of 3,000 tweets per second over several hours (corresponding to 259 million tweets per day).[1] At an order of magnitude higher, Facebook reported in 2009 that it was handling around 1 billion chat messages per day,[2] and there is widespread evidence of massive growth in web-based commenting systems (like on Reddit, Digg, and NYTimes) and other real-time "social awareness streams" [72].

Coupled with this explosion of short text in social media is the need for new methods to organize, monitor, and distill relevant information from these large-scale

[1] `http://blog.twitter.com/2011/03/numbers.html`
[2] `http://www.facebook.com/note.php?note_id=91351698919`

social systems. While some users may be interested in scanning over hundreds or thousands of short text resources, there has been a shift in recent years towards providing guidance to users to focus their attention on particular content. Several approaches exist for selectively focusing attention, including:

- Editorial selection: One approach is to rely on human editors to select representative short text. This is the approach taken by NYTimes which provides comment highlights: "A selection of the most interesting and thoughtful comments that represent a range of views." Editorial comments, however, may be biased toward the particular worldview of the comment selector and not representative of the themes of the comments themselves.

- Collaborative recommendation: In a separate direction, several sites allow users themselves to recommend content (e.g., through a thumbs-up/thumbs-down rating mechanism). For example Digg.com offers users the option of sorting short texts by the number of community votes to prioritize the content. Collaborative recommendations, while beneficial for aggregating a community's perspective, may not be very informative, favoring funny content or the content that are submitted by popular users. In addition, collaborative recommendations require adequate participation rates to be successful.

- Keyword Cloud: Rather than select particular short text, many websites support a keyword-based word cloud to show the most frequent topics and keywords used. For example, streamhacker.com, a famous blog about platforms, libraries, and languages, uses a tag cloud for each post. While keyword-based summaries may convey the overall flavor of a group of short texts, the keywords themselves lack the context and structure of a sentence-based comments for more detailed understanding.

While these and related methods provide a first-step toward making sense of the large amount of user-contributed short text in social media, the overall research goal of this dissertation is to develop the algorithms and methods necessary for ongoing information dissemination from such a large and growing body of content. While user-contributed short text offers the promise of a rich source of contextual information about Social Web content, it does so in a potentially "messy" form, considering the wide variability in quality, style, and substance of short text generated by a legion of Social Web participants. How can we make sense of the aggregate activities of millions of users? How can we spot high quality content that is preferred by web users? How can we organize short text so that users can easily find related concepts, opinions, and alternative viewpoints? How can we synthesize and represent the valuable content inherent in short text in social media, without burdening an editor with the onerous task of reading all possible short texts?

1.1 Overview of this Dissertation

With these questions in mind, this dissertation seeks to develop new algorithms and methods to ensure the continued growth of the Social Web by enhancing how users engage with short text in social media. Concretely, this dissertation takes a three-fold approach:

- **Rank.** First, this dissertation seeks to promote high-quality short text through *ranking*, so that low-quality short text may be filtered and user attention may be focused on highly-ranked short text. Such a ranking approach is challenging, however. Short text is inherently short, meaning it may lack the structure and attributes available for robust ranking (as in traditional Web content). Additionally, content quality may vary from community to community (e.g., NYTimes articles may attract insightful short text comments, whereas YouTube

comments may attract more juvenile ones), and so the ranking model should be flexible across these dimensions, so that short text quality is assessed relative to its community.

- **Label.** Second, this dissertation aims to organize short text through *labeling*, so that meaningful semantic descriptors may be assigned to short text for improved classification, browsing, and visualization. However, short text may contain terms linked to many possible semantic descriptors, leading to topic drift – e.g., a term occurring in short text like "state" may be associated with multiple, distinct labels, including mental states, states like Texas and Oregon, and other concepts not at all linked to the original short text. Additionally, the appropriate label may change over time, so careful determination of the temporal relationships between short text and candidate labels is important.

- **Summarize.** Third, this dissertation seeks to synthesize the important aspects discussed in a collection of short text through *summarization*. Unlike traditional multi-document summarization which has typically focused on high-quality documents in relatively small collections, short text summarization faces many unique challenges: e.g., high-variability in short text quality, wide ranging short text lengths (from one or two words to many paragraphs), multiple competing opinions, implicit references to earlier short texts, and so forth.

Concretely, this dissertation considers these three overlapping tasks in the context of three distinct sources of short text. We consider (i) comments, which are typically opinionated short text associated with a web entity like a news article or video; (ii) tweets, which are short text status updates posted on microblogging sites like Twitter; and (iii) reviews, which are short text assessing the quality and features of

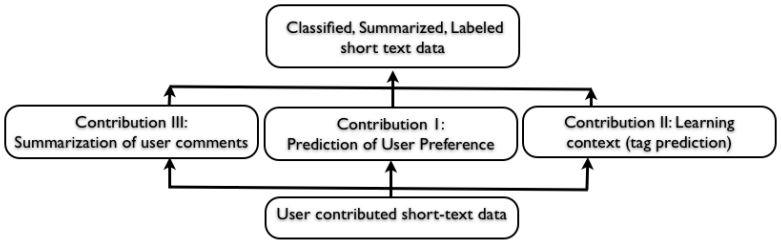

Figure 1.1: The three proposed contributions are used to classify and rank the user contributed short text based on user preference, label them based on the context and summarize them based on diversity and importance.

products listed on e-commerce sites. Towards answering the above challenges this dissertation makes three unique contributions (as illustrated in Figure 1.1):

- The first contribution is the development of a learning-based algorithm to automatically rank short text comments associated with a Social Web object (e.g., Web document, image, video) based on the expressed preferences of the community itself. By learning ranking functions for user-contributed comments, one could (i) automatically score new comments as they arise in the community; (ii) promote high-quality comments; (iii) filter out low-quality comments, so that user attention is not wasted; (iv) provide a sound basis for enhanced comment-based Social Web applications like summarization, content retrieval, visualization, and so on.

- The second contribution is a graph-based framework for automatically assigning relevant labels to short text. This framework links terms in short text to semantic annotations applied by users (via hashtags applied to Tweets). The method relies on a path aggregation technique for scoring the closeness of terms and candidate labels (hashtags) in the graph, so that high-value hashtags may be associated with Tweets, even if no terms have ever co-occurred with the hashtag. Such a label assignment approach is important for organizing short

text based on user-driven context, facilitating the retrieval of related short text, and improving how users access and visualize short text.

- The third contribution is a cluster-based summarization approach for extracting high-quality viewpoints expressed in a collection of short text, while maintaining diverse viewpoints. The proposed method identifies groups of thematically-related sentences from a collection of short text, ranks groups according to a measure of significance, ranks sentences within each group according to a measure of importance, and finally selects representative sentences from each group. By summarizing short text, user attention may quickly assess the aggregate viewpoints expressed in a collection of short text, without the need to scan each of possibly thousands of short text items.

The remainder of this dissertation is organized as follows:

- **Chapter 2** begins with the related work for this dissertation. It addresses the existing research and applications in the context of ranking, labeling, and summarization of short text in social media.

- **Chapter 3** studies the problem of ranking user-contributed short text based on the user preference. It introduces a learning based algorithm to automatically rank short text comments associated with a social web object based on the expressed preferences of the community itself.

- **Chapter 4** studies methods to understand the context of user-contributed short text in social media. It introduces a framework to recommend the suitable labels based on the latent relationships among terms in short text.

- **Chapter 5** introduces automatic summarization approaches of text and specifically the problem of summarizing user-contributed short text. This chapter

proposes cluster-based algorithms and criteria to evaluate the quality of a short text summary.

- **Chapter 6** concludes with a summary of the contributions of this work and provide a discussion of future directions that could build on the methods proposed in this dissertation.

2. RELATED WORK

2.1 Introduction

In this chapter, we examine work related to the three primary contributions of this dissertation. Specifically, we consider related work to the problem of identifying high quality content, understanding the context for label recommendation, and finally summarization of short text content in way that the gist of diverse viewpoints with high quality is reflected.

2.2 Ranking Short Text

A number of recent studies have examined challenges to the quality of user-contributed content, including the quality of user-contributed tags [83], blog comments [70], user-contributed answers on Question-Answering forums [2], product reviews on Amazon [40], and so forth. In many cases, these quality assessments rely on experts external to the Social Web community (e.g., a panel of human experts declares that a blog comment is "spam" or "not-spam"). Unlike many available studies we are interested in exploring how a Social Web community itself perceives the quality of user-contributed short text within the community, so that the community is the final arbiter of quality.

There are many studies about comments in message forums and newsgroups, including [25] and [71]. In particular, the Slashdot community – one of the acknowledged forebears of Digg and related social news aggregators – has attracted much attention. Several researchers have examined Slashdot's moderation policy for rating and filtering user-contributed comments, including [48] and [49]. Digg as one of the most successful social news aggregators among its rivals such as Reddit, Newspond, mixx5, Buzz!Yahoo, and SlashDot is different in a number of critical dimensions.

First, Slashdot offers a restricted form of comment rating (moderation) in which only a fraction of all users are selected to moderate a given comment. This restriction is in direct opposition to the Digg philosophy, in which all users are eligible to rate a comment. Second, Slashdot's comment rating policy restricts the ratings of a comment from -1 to 5, unlike Digg's comment rating system which is (potentially) unbounded, allowing for a wide variety of scores to be applied to comments. The structure of the Digg community could be potentially more problematic for sustaining the growth and quality of the community comment rating system – can the community really rely on the more democratic voting system in which all users can participate?

Lerman has studied Digg and its article rating system in some details, e.g., [52, 53, 54]. She has shown that users tend to like stories that were submitted by their friends and also were read and liked by them. This reveals that the social network behind Digg plays a significant role of promoting stories to Digg's front page, potentially leading to a tyranny of the minority situation in which a small number of interconnected power users have the most visibility and influence on the front page. However, to the best of our knowledge, there has been no previous work studying these users and their influence on comments on Digg, nor has there been any general study of Digg comments.

The impact of user contributed metadata such as tags, ratings and comments have been explored in several works. For example, some works used metadata to assist clustering online objects e.g [55, 78]. Works such as [14] studies methods to automatically generate personalized tags for web pages with the goal of easing the users digesting process of the rapidly emerging social opinion and information [28]. Among different metadata, comments have shown their special role in identifying web objects.

The nature of user comments is studied in [61] which considered term distributions of user comments to improve the search accuracy in a web search system. Other researchers [98] discovered that comments can further distinguish relevant objects from each other especially on popular objects where the comment set is large.

In this dissertation, we propose a learning-based model that can identify the important features relevant to each comment. This is through a training phase in which knowing the ground truth, the actual score of the comment received by the web community, will help us to understand the important features and their weights for a successful prediction. Using such features, we propose and evaluate a classification model and also a ranking approach for building a predictive model based on user preference.

2.3 Labeling Short Text

User-driven labeling is one of the organizing principles of most social media services – including image tagging (e.g., on Flickr), video tagging (e.g., on YouTube), and web page tagging (e.g., using Del.ico.us), among many others. And in these contexts, there has been considerable work in recommending tags.

In one direction, researchers have sought methods to aggregate the collective knowledge of web users to expand the small set of tags applied to a resource with other user-contributed tags [85, 87, 94, 26, 69, 23]. In such collaborative filtering based approaches, the number and frequency of tag co-occurrence builds the core model of tag recommendation. Given a set of tags already input by the user for a new resource such as a picture, URL, or a blog post, these algorithms suggest new tags based on the number of co-occurrences of such input tags with the previous annotations.

In a different direction, other studies have recommended personalized tags for

each user based on the user's history, bookmarks, and other personal documents. They have proposed a collective based tag recommendation, P-TAG, which is an automatic method to generate tags for web pages using the textual content of target page as well as the documents residing on the surfer's desktop [14].

In one study [77] they have used the knowledge residing in three different contextual layers in a probabilistic framework. The contexts were personalized, social, or collective which results in considering the co-occurrences seen in person's history, friends network or all the network respectively. They have concluded that while recommendations that are based on collective knowledge makes good recommendations on a large number of users with diverse interests, it is possible to miss some recommendations that are particularly relevant in a personal context. However personalized recommendations provide good results for the active users with known levels of interests and those users who are conscientious while annotating. Since such users cause the statistics underlying the recommendation system reliable.

There are many studies on the usage of data mining approaches (like association rules) to predict the appropriate tags for content-rich resources such as webpages [29, 93] in which [29] discusses the fact that tuning the recommendation system for high recall will naturally encounter a decrease of precision. However it is desirable to have a high recall to link disparate vocabularies among web users.

Several efforts have focused on graph-based approaches, in which the relationships among tags, resources and users are modeled as a tri-partite graph [39]. In such settings, important "power" tags, users, and resources may be identified through the application of a PageRank-like iterative algorithm. Similarly a tag-document bipartite graph has been used as the basis to cluster tags and documents, as discussed in [86]. To recommend a tag for a particular document, they first identify the cluster of a document and then a Poisson mixture model is applied to rank the tags in the

selected cluster. This method is based on receiving different tags for one resource.

In addition to these efforts, there have been many other approaches for tag recommendation [68, 36, 9, 96, 100]. Association rules have shown encouraging results for unifying different languages of a term and the super-subconcept relationships.

There are many studies about spreading the tags on Twitter network defined by the interactions among Twitter users [80]. They found significant variation in the ways popular hashtags on different topics spread. Also they discussed the probability of adoption of a tag by online users, and how rapidly the usage of such tags decays during time. Many studies [85, 77, 23] have introduced interactive recommendation algorithms in which given a set of tags already input by the user for a new picture, the algorithm can then predict a new tag based on the number of co-occurrences of such input tags with the previous annotations. In another place Bayesian principle [94] has been used for tag recommendation which again considers the co-occurrence among the tags for the web objects. Another work considers the similarity of the annotated webpages and the similarity among tags to expand the input tag set [59]. Many works have studied tag suggestion, from a collaborative filtering and UI perspective, for example with URLs and blog posts [69, 96, 87]. Most such methods can not be applied to short text data, since there is no relevant text, URL or input tags available to help us suggest a suitable tag. There are many studies on the usage of association rule to predict the appropriate tags for a content rich resources such as a webpage [29, 93]. Increasing the recall of predictions is the main focus of such studies even though a decrease in precision is encountered. The success of such systems is defined as retrieving more resources for a query. Graph-based ranking algorithm was proposed by many studies in which both the relevance to the document and preference of the user is taken into account [27, 39, 86]. All of which is based on receiving different tags for one resource . One approach is to

convert a folksonomy to an undirected tri-partite graph with nodes for tags (T), resources (R) and users (U); a Pagerank algorithm is applied to it to highlight the powerful users, power tags and power resources [39]. A clustering and then classifying framework [86] was developed for tag recommendation in which a spectral clustering was applied on the bipartite graph to simultaneously group tags, documents and words into clusters. Then a two-way Poisson mixture model was trained on the obtained clusters. Given a query document, the algorithm computes which cluster they belong to and then a ranking was applied on the tags in that cluster. Other work analyzed the ways in which tags spread on Twitter network defined by the interactions among Twitter users [80]. They found significant variation in the ways popular hashtags on different topics spread. They introduced metrics to show the probability of adoption of a tag by online users, and how rapidly the usage of such tags decays during time. Overall, recommending tags can serve various purposes, such as: increasing the chances of getting a resource annotated, reminding a user what a resource is about and consolidating the vocabulary across the users [39]. A tag recommendation module can assist users in the tagging process by suggesting relevant tags to them. It can also be directly used to expand the set of tags annotating a resource. The benefits are twofold: improving user experience and enriching the index of resources [27]. It also helps us to gain insights into the "information content" of tags used in the social tagging systems. Tag prediction is used as a recall enhancing device of the tag feeds. It facilitates the sharing process of online objects despites of vocabulary differences. It also helps with disambiguating what a user meant when annotating an object. The way users use tags is determined by previous experience with tags in the system. Tag prediction can pre-seed a system with appropriate tags to encourage quality contributions from users [29]. Being able to effectively recommend tags would, firstly, simplify the tasks of the users on the web who want to

tag resources such as bookmarks, pictures, and, secondly, would allow an automatic annotation of resources that enables, for example, a better search for resources or an improved resource recommendation [37].

Like much of the related work on adding semantic descriptors as labels to the web entities, we also are interested to provide a potentially scalable mechanism to organize the web as it continues to grow. Indeed we would like to suggest relevant labels in a real-time manner in way that reflects the more temporary events and at the same time captures the context of short text. Compared to these related efforts, automatically assigning labels to short text differs in three fundamental ways. First, in traditional social media tag prediction, the tagged resource itself (e.g., the video, the image) is typically made available for collaborative tagging. That is, an image on Flickr may attract dozens or hundreds of contributors who provide their own tagging perspective on what the image is, providing a rich source of tagged information for a single image. By comparison, a status update on the real-time web is annotated by just one user and typically with only one hashtag, meaning there is not a rich collection of collaboratively shared hashtags available to describe a single status update. Second, for the purposes of hashtag prediction, the status update itself is a sparsely described object. Most status updates are short (as on Twitter, where there is a 140 character limit) and so there is little evidence in the status update itself; in contrast, web pages and other social media often contain richly available descriptive evidence (e.g., in the text of the page itself) to augment tag prediction. Third, the real-time web is necessarily a rapidly evolving medium, with millions of updates per day and highly-dynamic tagging behavior, meaning that the tags themselves may rapidly evolve and change in use and purpose (as compared to a Flickr photo of a well-known landmark, in which the tags associated with the landmark are typically much longer-lived and less dynamic). Hence, it is important

to develop a new approach for automatically assigning labels to short text on the real-time web.

2.4 Summarization of Short Text

The history of automatic text summarization goes back to 1958 where researchers suggested that text summarization by computer was feasible though not trivial [62, 6, 18]. These original algorithms were based on sentence position [62, 6] and word frequency count [18] to select portions of the input text as extractive summaries. Many years later with the advent of the web and large set of online corpora the interest for automated text summarization renewed. New advancement on Natural Language Processing (NLP) and Information Retrieval (IR) techniques plus computers with higher speed and larger memories made more sophisticated algorithms feasible. [32]. Years later machine learning techniques were applied on a set of natural language processing features to identify the important key part of the input text as a summary. The pioneers were [47], who developed a summarizer using a Bayesian classifier to combine features from a corpus of scientific articles and their abstracts [50] and [32] who experimented with other forms of machine learning and its effectiveness. Machine learning has also been applied to learning individual features such as sentence position [56], important words and phrases and their syntactic context [95]. Hovey talks about available of summarization methods as the following [32]:

> "when one takes a moment to study the various systems and to consider what has really been achieved, one cannot help being struck by their underlying similarity, by the narrowness of their focus, and by the large numbers of unknown factors that surround the problem."

We do not see much of a difference of summarization methods unless we consider different domains in which there are properties that need to be considered based on the intrinsic properties of the input text. There is no exact agreement on the definition of summary and each person interpret it based on his need. The summary is defined by Hovy as the following [32].

> "A summary is a text that is produced out of one or more (possibly multimedia) texts, that contains (some of) the same information of the original text(s), and that is no longer than half of the original text(s)."

Some research uses the degree of lexical connectedness between potential summary and the remainder of the text. Connectedness is measured by the number of shared words or synonyms [81, 63, 5].

Another effective approach is to reward sections of the input text that include topic words, that have been determined to correlate well with the topic of the source text [74]. Furthermore they have developed an open-source summarization environment, MEAD, that allows researchers to experiment with different features for an effective summarization.

Some work [15] has turned to the use of hidden Markov models (HMMs) and pivoted QR decomposition to reflect the fact that the probability of inclusion of a sentence in an extract depends on whether the previous sentence has been included as well.

To automatically summarize user-contributed short text through a process of identifying and extracting key informative content we are inspired by recent efforts at automatic text summarization for creating a compact version of either a single document or a collection of documents [19, 76, 67]. In the short text ranking method we want to pick up the sentences that are playing the role of summary or abstract

of all of the input short text data. Therefore, we look at the first few short text in our ranking as the summary of all input short text.

There are studies to provide summaries in the format of the Tag cloud by considering the hidden relationships in the underlying content of the comments [46, 45]. With the use of tag cloud, some of those relationships have been discovered more efficiently. The criticism we make for it is that the tags alone are not reflecting the context. We need to have sentences to understand the main idea of a document.

In a study [4] the metrics for document summarization are developed. They first found the relevant sentences in a document and then applied novelty measures to filter the redundant sentences from the collection. There are works on using comments to summarize a document such as a blog [35]. It was showed that the terms appearing in the comments are a good pivot of sentence importance in an article. To discover different aspects of the objects on the web, [60] used user reviews. They have extracted different aspects of a product like a car such as mileage, engine, transmission, and extracted the crowd idea about each of such aspects. This works great with controlled vocabulary seen in e-commerce website such as amazon.com.

Many studies apply graph based ranking for single or multi document summarization and they select the top-K sentences as the summaries of the input document(s). Example includes TextRank [66], LexRank [19]. Similar to Google's PageRank algorithm [73] or Kleinberg's HITS algorithm [43], these methods first build a graph based on the similarity relationships among the sentences in a document. MEAD and LexRank methods have shown good results for single or multi documents summarization giving some pages of concrete articles. The position of the sentences and the similarity of the sentences to the title of the resource are among features that are used. Three default features that come with the MEAD distribution are Centroid, Position, and Length. A centroid is a set of words that are statistically important to

a group of documents. This can be identified by considering the words with TFIDF greater than a threshold. Position is the normalized value of the position of a sentence in the document such that the first sentence of a document gets the maximum Position value of 1, and the last sentence gets the value 0.

Here we introduce different variations of a summary. Any summary can be characterized by at least three major classes of characteristics. Input,output and purpose. This categorization is also mentioned extensively in [75, 32]. The input of the summary could be characterized as one of the following:

- single-document vs. multi-document: single document is a single input text while for multi-document there are more than one input text that is thematically related. We consider several short text as multi-documents that are related to each other.

- domain-specific vs. general:The input text could be from a specific or different domains. Since we consider the short text for one web object, our method is for domain-specific case. Therefore it is appropriate to apply domain-specific summarization techniques with less term ambiguity, focus on specific content.

- Genre and scale: Typical input genres include newspaper articles, newspaper editorials or opinion pieces, novels, short stories, non-fiction books, progress reports, business reports, and so on. The scale may vary from book-length to paragraph length. The scale in our case is all the short text pertaining to a single resource.

The output of a summary is characterized by the following criteria.

- Extract vs. abstract: Extractive summarization selects representative text segments, usually sentences, from the original documents. An abstract is a

newly generated text, does not use the existing sentences from the input data, it analyzes documents and directly generates sentences. Since it is difficult to generate readable and complete sentences, studies on extractive summary are more popular than that on abstractive summary. The methods proposed in this dissertation is also extractive.

- Fluent vs. disfluent: A fluent summary is written as a full, coherent structure. A disfluent summary is fragmented, consisting of individual words or text portions that are not composed into coherent paragraphs. We deal with disfluent summaries in this dissertation

- Neutral vs. evaluative: A neutral summary reflects the content of the input texts, partial or impartial as it may be. An evaluative summary includes some of the system's own bias by including the sentences with biased opinion. We take the neutral path in this dissertation.

- Fixed vs. floating: A fixed summary is created for a specific use, reader (or class of reader), and situation. A floating situation summary does not assume fixed conventions, but is created in a variety of settings to a variety of readers for a variety of purposes. We propose floating summaries in this dissertation.

The purpose of the summarization may be one of the following, depending on the use made of the summary.

- Generic vs. query-oriented: A generic summary is for all types of audiences. It provides the author's point of view with equal inclusion of all major themes. A query-oriented summary is based on user need and it favors specific aspects that is important for user's desire. In this dissertation we are dealing with generic summarization.

- Indicative vs. informative: An indicative summary provides an indication of the core subject of the input texts without including its contents. An informative summary reflects some of the content of the the input text. Our case is informative.

- Background vs. just-the-news: A background summary assumes the reader's prior knowledge of the general setting of the input texts content is poor, and explains background information such as circumstances of place, time, and actors. A just-the-news summary contains merely the new or principal themes, assuming that the reader knows enough background. In this dissertation we do not intend to provide any background so the proposed method is categorized as just-the-new.

One variation of the general topic of summarization is called *opinion summarization*. It considers a different aspect of the input data such as individual features about a resource accompanied by the polarities pertaining to each. The process includes separating input data by polarities and topics to generate the most representative text snippet from each topic. This type of summarization has been used in reviews about different resources such as a book, movie, or any type of products. The problem of opinion summarization itself could be categorized into *aspect-based summarization* and *non-aspect-based summarization* [41].

Some studies in aspect based summarization have 3 clear separation steps for summarization; Aspect/Feature Identification, Sentiment Prediction, Summary Generation. [33, 34] In contrast some of them do not have such a clear separation. They are referred as *integrated approaches* [41]. Examples include [65, 91] which mainly use probabilistic mixture models namely PLSA [30] and LDA [7].

The non-aspect-oriented summaries on the other hand produce a generalized summary without consideration of aspects. Such approaches share similar concepts from text summarization [58, 42, 20, 12]. For example Contrastive summaries [42] generate two sets of sentences; positive and negative. There is no notion of classification of the input text into separate features. Another example of non-aspect-oriented summaries is Abstractive summaries in which new content is created from the available input sentences. The abstractive summarization method, Opinosis [20] generates graphs of words and based on the part of speech (POS) tags, the meaningful paths are identified as the summary. While in another abstractive method [21] an unsupervised optimization approach is used to generating ultra-concise summaries of opinions.

3. RANKING COMMENTS: A COMMUNITY-PREFERENCE APPROACH*

3.1 Introduction

In this chapter we propose and evaluate a regression-based learning approach for automatically ranking comments based on the expressed preferences of the community itself. Concretely, we study the popular social news aggregator Digg and the socially-generated comments that Digg users can annotate news articles with. We explore several factors impacting the community's preference for user-contributed comments, including the contributor's reputation and community activity level, as well as the complexity and richness of the comment. Knowing that content posted earlier may receive more social attention [89], it is important when training a ranking model, to balance the visibility of the comment with its intrinsic quality (i.e., breaking the feedback loop, so that early comments are not always preferred over later comments). Finally, it is important that a ranking model perform especially well on the top-k comments for small k, since users and applications are typically most interested in these high-quality comments.

3.2 Background

Commenting systems on the Social Web have been growing in popularity in the past few years, from blogs and social media sites like YouTube and Flickr to major news sites like NYTimes.com. Many of these commenting systems include a rating component, so that users can rate the comments submitted by other users.

In this chapter, we study Digg, a popular social news aggregator. Digg users can submit stories to the community, rate stories that have been submitted by others

*Reprinted with permission from "Ranking Comments on the Social Web" by Chiao-Fang Hsu, Elham Khabiri and James Caverlee, 2009. *Computational Science and Engineering*, 4, 90-97, Copyright 2009 by IEEE.

Figure 3.1: Example article with 315 "Diggs" (article community rating).

(to "Digg" a story is to cast a positive vote for it), comment on stories, and engage in other typical Social Web activities (e.g., make friends with other users, track the stories that have been "Dugg" by others, and so on). With more than 27 millions visitors in the past year, Digg is one of the most successful social news aggregators, and an even more popular Web destination than NYTimes.com and CNN.com [according to statistics from compete.com].

Figure 3.1 illustrates an example submission to the Digg community, in this case a news article about Kathleen Sebelius. We can see that the story has received 315 Diggs (or 315 positive votes) by members of the Digg community, making this a quite popular story. We call this score the *article community rating*. Figure 3.2 illustrates a sample comment associated with this article. Digg users may rate each comment using a simple thumbs-up or thumbs-down rating system; in this case, the comment has an aggregate score of 37 (45 up-votes and 8 down-votes). We refer to the aggregate score assigned to a comment as the *comment community rating*. When considering all of the comments associated with a Social Web object, we can order the comments according to these community-based ratings; we refer to this as the *community preference* for these comments. Our goal is to develop automatic techniques for learning this *community preference* even in the absence of explicit

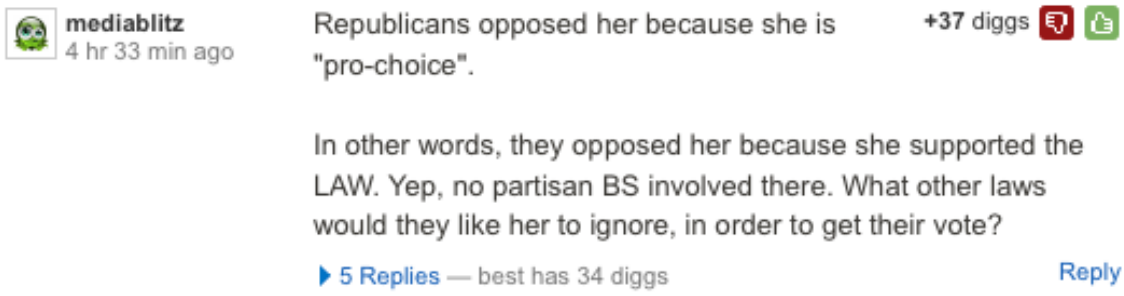

Figure 3.2: Example comment associated with the article in Figure 3.1. This comment has a comment community rating of +37.

community ratings.

Figure 3.3 shows the distribution of community ratings for all of the comments harvested from Digg. Note that the majority of comments receive an aggregate positive score, though with some outliers at both the extreme negative and positive ends. The maximum comment score is 2357, the minimum is -861, and the mean of comment score is 2.

Comments on Digg range in style and perceived quality within the community; some examples include the informative and highly-rated (like the comment in Figure 3.2 with 223 up-votes and 8 down-votes), to the humorous (e.g., "**** the RIAA"), to the poorly received (see Figure 3.4).

Figure 3.5 shows the first 4 comments submitted for story of Figure 3.1. The first comment had the chance to show itself to many users and hence received a large number of diggs. In this case +338 diggs. However the second one was not liked by the digg community and has received a total score of -14 diggs. Comparison between the comments that are sorted by time Figure 3.5 and the comments sorted by Digg score Figure 3.6 reveals that earlier comments has more chance to be dugg by the community and there is a strong overlap between position and the number of received thumbs up. As it is shown in Figure 3.7 the most recent comments

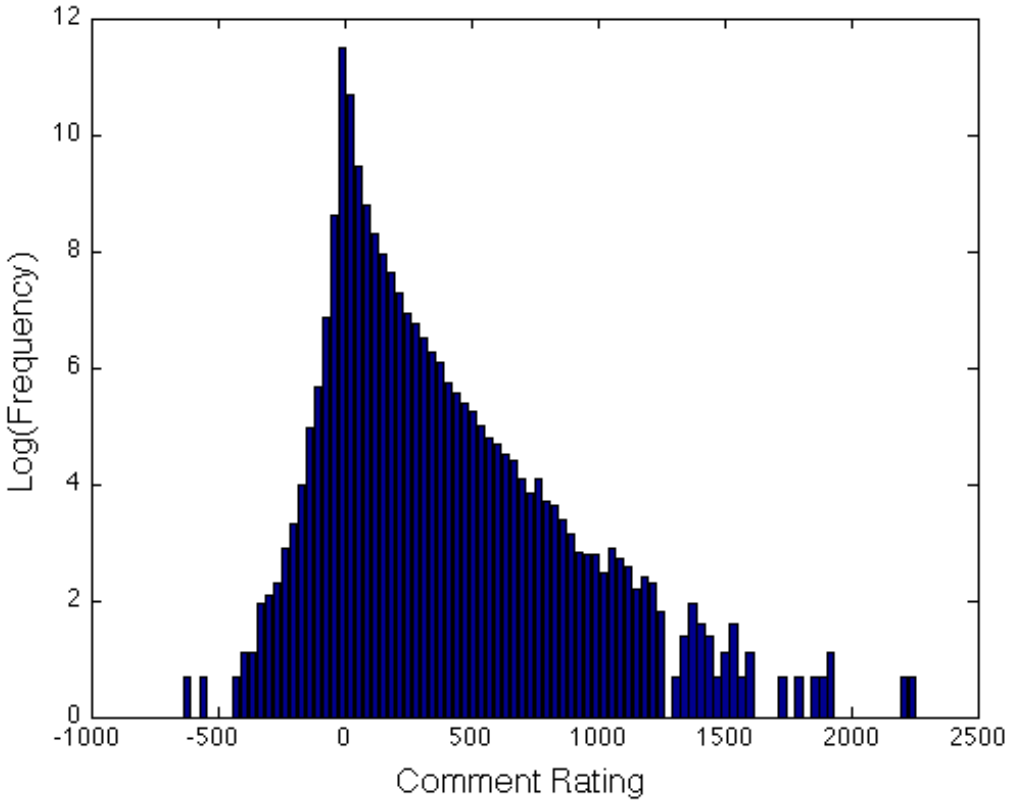

Figure 3.3: Distribution of comment community ratings for the Digg dataset.

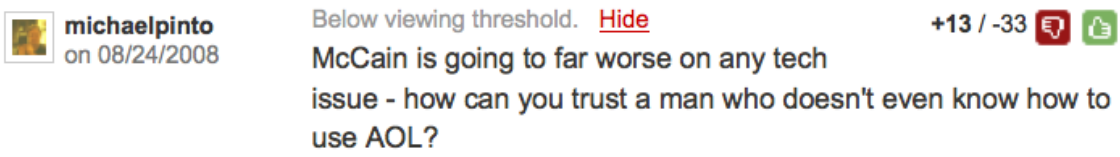

Figure 3.4: Example lowly-rated comment on Digg.

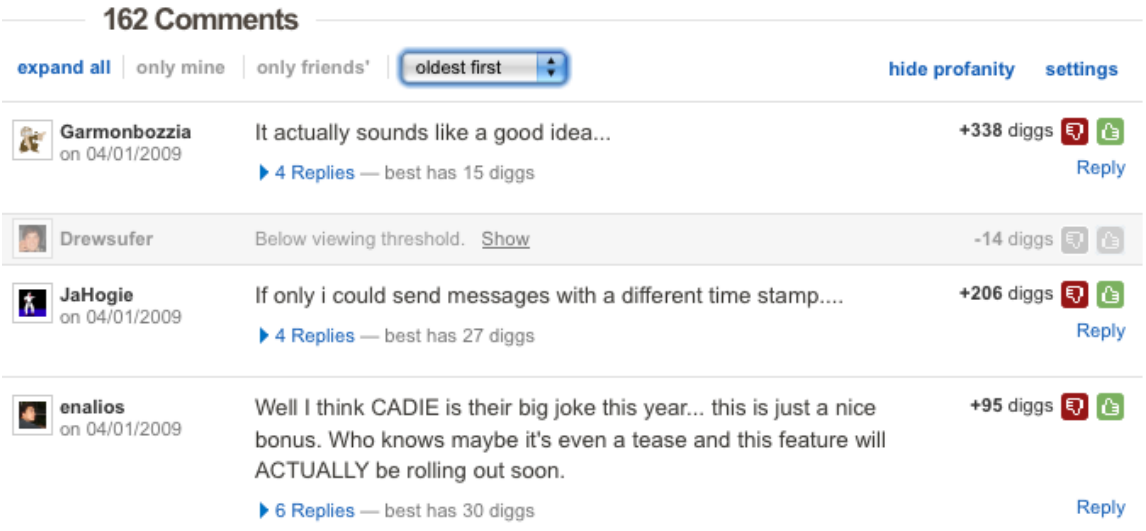

Figure 3.5: Comments sorted by time (oldest first)

received few votes and the average of the votes decreased drastically. This means that some recent comments with more interesting content are buried by the earlier comments. Later we will propose a method to lighten such biased judgement based on the position of a comment.

3.3 Classification of Comments

Based on our analysis of Digg community preference for comments, we propose a learning-based approach for predicting the community's preference rating of unseen comments.

3.3.1 Prediction Framework

The prediction framework relies on a classification approach for building a predictive model. The goal is to predict for an unseen comment one of four different labels: Excellent, Good, Fair, and Bad. Recall Figure 3.3, where we plot the distribution of comment ratings in our Digg comment dataset. In our experiment, we define the class boundaries such that a comment with the score of less that -100 is

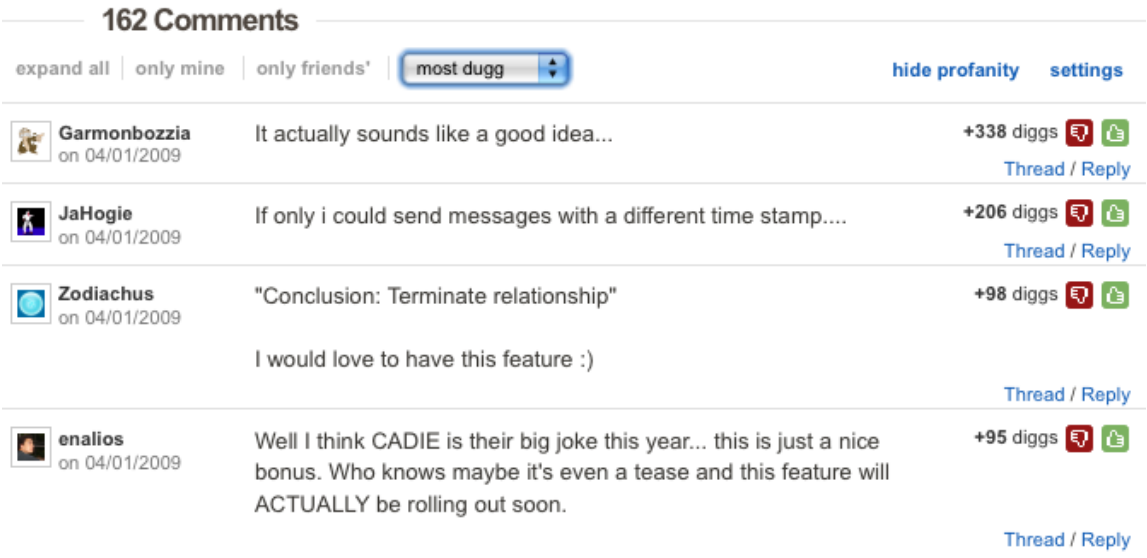

Figure 3.6: Comments sorted by Digg score

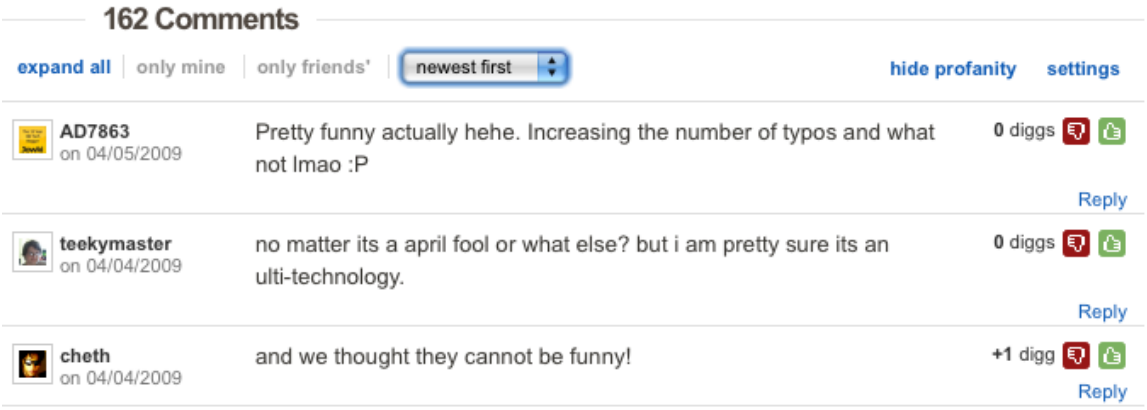

Figure 3.7: Comments sorted by time (newest first)

considered as a Bad comment. Comment score between -100 and 0 is Fair, between 1 and 600 is Good and greater than 600 is Excellent. We train two different classifiers over 90% of the comments to build the model using the features described in the previous section. We evaluate the quality of the model over the held-out 10% of the comments. Concretely, the two classifiers used in here are Linear regression and Quadratic classifier.

Linear regression Classifier: The relationship between the features is modeled by fitting a linear equation to the ground truth which is the Digg score of the comments. Each feature will receive a weight based on the influence it shows on the training data.

$$\sum_{i=1}^{15} f_i * w_i = S \tag{3.1}$$

where f_i is feature i, w_i is the weight of feature i and S is the Digg score vector of the comments. Through the training process we first obtain the regression weights. Later we apply the learned weight to predict the score for the test samples.

Quadratic Classifier: In a quadratic classifier the posterior probability of each class is evaluated and the class with the largest $P(w_i|x)$ is selected. That is, knowing x as a comment, what is the probability of its membership in class w_i. The class with the highest posterior probability will be assigned to the test sample. With the assumption of a Gaussian distribution of the samples the following quadratic equation is used:

$P(w_i|x) =$

$$-\frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i) - \frac{1}{2}log(|\Sigma_i|) + log(P(w_i)) \tag{3.2}$$

Here μ_i and Σ_i are the mean and covariance of each training class w_i. The prior

probability $P(w_i)$ is selected based on the percentages of training set comments in each of these categories.

Method	Rate	Precision	Recall
Reg.	80%	0.01 0.38 0.64 0.00	0.03 0.02 0.94 0.14
Quad.	85%	0.25 0.82 0.70 0.02	0.03 0.23 0.96 0.04

Table 3.1: Precision and recall for different classification methods

3.3.2 Preliminary Results

In our initial evaluation, we measure the classification rate, precision, and recall over the test set of comments. The classification rate measures the percentage of the comments that were classified correctly. The precision and recall was calculated for each group (Bad, Fair, Good, Excellent) separately. In the Excellent group for example the Precision is the number of actual excellent comments (true positive) retrieved by a our system divided by the total number of retrieved Excellent comments by our system: $Precision = \frac{TP}{TP+FP}$. Table 3.1 reports the evaluation measures for the two classifiers using the base set of boundary values. We find that the quadratic classifier approach has a higher classification rate as well as higher precision and recall in most groups. We see that the precision for the Fair and Good categories is high (0.82 and 0.70) relative to the precision for the Bad and Excellent categories (0.25 and 0.02). These latter two categories are relatively small and difficult to predict. We are encouraged, however, by the success in differentiating between Fair comments and Good ones.

3.4 Learning to Rank Comments

In this section, we present the formal model for ranking comments on the Social Web by community preference. We approach the problem of ranking comments as

a regression problem. Consider a set of k Social Web objects (e.g., Web documents, images, videos) $O = \{o_1, o_2, ..., o_k\}$. Each object o_i has a set of up to n comments associated with it $C_i = \{c_{i1}, c_{i2}, ..., c_{in}\}$. Each comment c_{ij} has a set of m features $F_{c_{ij}} = \{f_1, f_2, ..., f_m\}$. Each feature refers to some quality measure with respect to the comment. In the following section, we will explore a number of different possible feature choices. We assume there exists some training data that has the form:

$$\{(F_{c_{1,1}}, r_{c_{1,1}})...(F_{c_{1,n}}, r_{c_{1,n}}), (F_{c_{2,1}}, r_{c_{2,1}})...(F_{c_{2,n}}, r_{c_{2,n}}), ...,$$
$$(F_{c_{k,1}}, r_{c_{k,1}})...(F_{c_{k,n}}, r_{c_{k,n}})\} \subset F \times \mathcal{R}$$

where the pair $(F_{c_{ij}}, r_{c_{ij}})$ corresponds to the feature set for comment c_{ij} and the comment community rating $r_{c_{ij}}$ for comment c_{ij}. To tackle the community preference-based ranking problem, we can train a regression model over this training data. Concretely, we build the model through (i) a selection of features, as we will discuss in the following section; and (ii) the application of Support Vector Regression [17], a state-of-the-art regression model similar-in-spirit to the popular Support Vector Machine classifier that has proven successful across many domains, e.g., [82]. Support Vector Regression uses an ϵ-insensitive loss function that defines a tube with radius ϵ around the hypothetical regression function. If the data is placed within this tube, the loss function can be regarded as 0. By introducing the positive slack variables ξ_i and ξ_i^*, the SVR regression can be formulated as the constrained optimization problem:

$$Minimize \quad \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i + \xi_i^*$$

$$Subject\ to \begin{cases} r_i - w^T\phi(F_{c_{ij}}) - b \leqslant \epsilon + \xi_i \\ w^T\phi(F_{c_{ij}}) + b - y_i \leqslant \epsilon + \xi_i \\ \xi_i, \xi_i^* \geqslant 0, i = 1, ..., l, \epsilon > 0 \end{cases}$$

where $\phi(F_{c_{ij}})$ is the feature mapping for each comment in the high dimensional feature space, w and b are the slope and offset of the regression line, and $C > 0$, called the regularization parameter, is a positive constant. The positive slack variables ξ_i and ξ_i^* are to measure the deviation of training samples outside the tube ϵ zone. The constrained optimization problem given by the equation can be reformulated into a dual problem formalism by introducing Lagrange multipliers. Based on the Karush-Kuhn-Tucker conditions, the function is given by:

$$f(F_c) = \sum_{g=1}^{k*n} \sum_{h=1}^{k*n} (\alpha_g - \alpha_g^*) K(F_{c_g}, F_{c_h}) + b$$

where α_g, α_g^* are the Lagrange multipliers corresponding to the training data. Note that for those comments that serve as support vectors, the $\alpha_g > 0$ and $\alpha_g^* > 0$ whereas all the other comments must have $\alpha_g = 0$, $\alpha_g^* = 0$. $K(F_{c_g}, F_{c_h}) = \phi(F_{c_g})\phi(F_{c_h})$ denotes the kernel function, which satisfies the Mercers conditions. The kernel function we used in this work is the radial basis function: $exp(\gamma * |F_{c_g} - F_{c_h}|^2)$. In practice, we use a robust SVR implementation with default parameters available as part of the LIBSVM package [11]. In the testing phase we use this model to predict a rating for the unseen comments associated with an object $S = \{s_1, s_2, ..., s_n\}$ (e.g., $S = \{30, 100, 40\}$). Based on these ratings we can determine the relative rank order of the unseen comments: $R = \{r_1, r_2, ..., r_n\}$ (e.g., $R = \{3, 1, 2\}$). Note that our goal here is not to precisely estimate the actual comment community rating for a comment. Since comments may be continually rated, a predicted rating may quickly

become stale. Instead, our goal is to predict the *relative order* of comments, so that even as new ratings are made on the comments, the model will be able to capture the relative quality.

3.5 Comment Representation

Given the baseline ranking model, we now turn to the choice of features to represent the comments. The quality of a ranking model is strongly influenced by the quality of the features used to model the domain. In this case, we study several factors that we hypothesize may influence comment community ratings – the visibility of the comment, the influence and reputation of the user contributing the comment, and the content of the comment itself. Note that although the following discussion focuses on Digg for clarity, the proposed model is designed for use with any collection of Social Web comments.

3.5.1 Comment Visibility

The first factor we consider is comment visibility within the community. Intuitively, if more users in the community view a comment, it is more likely to attract a larger community rating. Conversely, a comment that is viewed by very few community members (say, for a comment related to an article that is of little interest to the community), will have less capacity to attract a large community rating. We measure the visibility of a comment through two factors: (i) the *article community rating* of the article that the comment is attached to; and (ii) the *comment posting time*, since earlier comments may have the capacity to be viewed by more community members than later arriving comments. Figure 3.8 shows the average article community rating versus the average comment rating (for the top-50 comments per article) across eight top-level Digg categories. The correlation coefficient is 0.93, validating the intuition that articles with high visibility (via many article Diggs) attract more

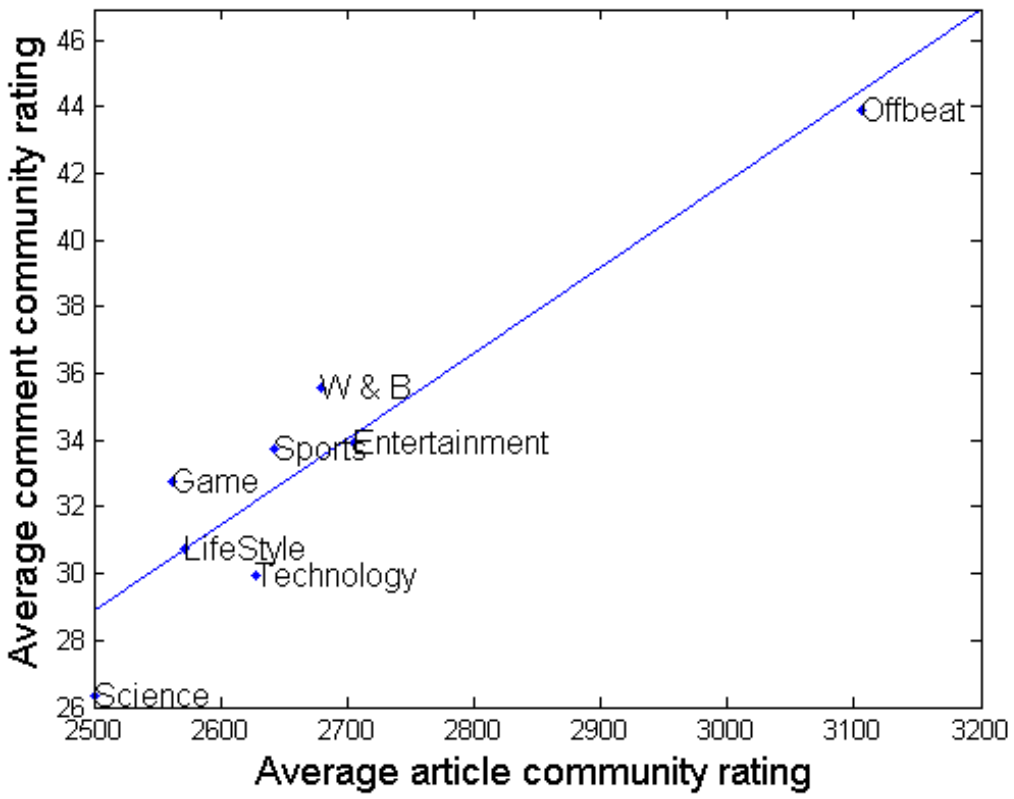

Figure 3.8: Avg article rating vs. avg comment community rating (by category).

votes for their comments.

Figure 3.9 shows that the mean score of comments that arrive earlier is greater than the mean score of comments arriving later, though with greater variability for early comments. In the figure, comments are arranged in order of their posting time (e.g, 1st, 2nd, ...). An early comment has greater visibility, and hence, greater capacity for a high community rating.

Recall that our overall goal is to automatically find the relative rankings of the comments associated with an article, even in cases when the community has not yet made its aggregate community preferences known. Hence, the first visibility feature (article community rating) will not necessarily be available for our prediction

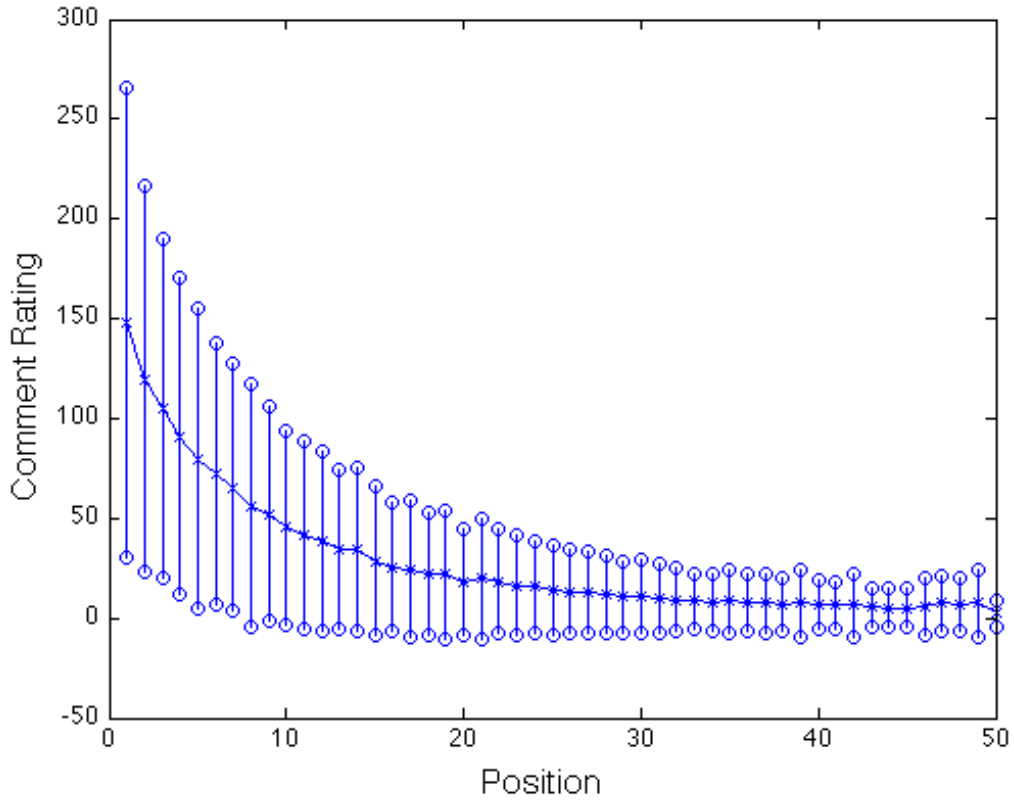

Figure 3.9: Comment posting time (by position) versus comment community rating. We report the mean comment rating +/- one standard deviation.

framework. As a result, we train the regression models with the article community rating feature to control for the article visibility bias across articles. For the testing phase we ignore the article community rating since it may not be known in practice and since all comments for an article would share the same feature value.

The second visibility feature – comment posting time – is known in the testing phase, and so we can use it as a prediction feature. Of course, it may be reasonable to try to control for comment posting time in much the same way we have controlled for the overall article visibility – so that potentially high-quality comments that happen to arrive late (and hence, may receive a low score due to low visibility within the community) are boosted to a higher position. Indeed, we study one possible "correction" factor in Section 3.6.5.

3.5.2 User Reputation and Influence

We consider reputation and influence of the user contributing the comment. We want to know if a power user's comments will be more interesting and valuable to the Digg community. Here are some per-user features.

The first set of user-based features gives insight into each user's activity and interest level within the community:

- *Number of articles submitted*: This measures a user's activity in the community by the number of articles the user has submitted to the Digg community.
- *Community membership date*: This feature indicates how long the user has belonged to the community. For smoothing purposes, the account starting date (yyyymmdd) of each user is normalized into the range of 0 to 1, with higher values indicating newer members.
- *Category activity*: We calculate the percent of that user's article ratings to

articles from each of the eight top-level Digg categories (e.g., Sports, Technology). For a comment from this user on a particular article, we take the user's category activity percentage for the article's category. The intuition is that for users who comment in an area of their expertise, their comments may have a higher likelihood of being appreciated by the community.

The second set of user-based features measures user popularity in the community:

- *Number of articles appearing on the Digg front page*: Digg uses a proprietary promotion algorithm to determine which stories submitted by its users reach the front page of Digg (and hence, reach the largest audience). A user who has had success submitting stories that reach the front page is an influential member.

- *Number of profile views*: How many times has the commenter's Digg profile been viewed?

- *Number of friends*: The number of friends of the commenter is recorded. Users with many friends may be more appreciated as commenters.

The final set of user-based features considers how well each user has participated in commenting in the past:

- *History of received comment ratings*: This feature measures the aggregate (sum) rating of a user's past comments. Does this user tend to make highly-rated comments? Or lowly-rated comments?

- *History of received comment replies*: This feature measures the number of replies that the commenter has received from his past comments and can be viewed as a reflection of how much his comments have been interesting.

3.5.3 Content-Based Features

The third factor we study are features related to the content of the comment itself. Since Digg and other Social Web websites attract comments from users with a wide-range of educational backgrounds, ages, and interests, the comments these users contribute may vary greatly in word choice, grammar, use of novel phrases, and so on. To capture the impact of these content-based attributes, we consider several semantic and statistical features of the comment text.

The first set of content-based features reflect some statistical properties of the text:

- *Comment length*: The first feature measures the number of words in the comment text. There may be a tradeoff between longer comments compared with the community's time and effort spent to appreciate the comment. We hypothesize that the Digg community values average-length comments rather than extremely short or extremely long comments. Although a long comment may be more informative, the community may not appreciate the effort to read and understand it. Studying the relationship between comment score and its length, we found that the comment score is maximum for short comments.

- *Comment complexity*: We measure the complexity of a comment by the entropy of the words in the comment. The entropy of a comment reflects the richness of the comment by measuring the variety of words in the text. In our experiments we found that comments with less complexity get higher Digg scores. Equation 3.3 shows that for a comment c_j with λ number of words

what is the entropy of c_j when each word has frequency p_i.

$$entropy(c_j) = \frac{1}{\lambda} \sum_{i=1}^{n} p_i [log_{10}(\lambda) - log_{10}(p_i)] \tag{3.3}$$

- *Number of upper case words:* This is a simple count of upper case words.

- *Comment informativeness*: Informativeness is meant to capture the uniqueness of the content in a comment relative to other comments attached to the same Social Web object. We measure the informativeness of comment c_j using a variation of the standard TFIDF approach from information retrieval, where we sum over the TFIDF values for all terms in a single comment:

$$inform(c_j) = \sum_{t_i \in c_j} \mathrm{tf_{i,j}} \times \mathrm{idf_i}$$

  The tf component values terms that occur frequently within a comment: $tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$ where $n_{i,j}$ is the number of occurrences of the considered term in comment c_j, and the denominator is the sum of number of occurrences of all terms in comment c_j. The idf component values terms that occur infrequently across comments $idf_i = \log \frac{|C|}{|\{c:t_i \in c\}|+1}$ where $|C|$ is the number of comments and $|\{c : t_i \in c\}|$ is the number of comments in which t_i appears.

- *Category cohesion*: This feature measures the commenter's word choice with respect to the other comments within a particular category. The hypothesis is that each category has its own sub-community that uses particular jargon. Hence, comments that have high cohesion with the rest of the category are more likely to receive high ratings. We measure category cohesion using the sum of the Mutual Information (MI) between all terms in the comment and

the category (cat) of the article:

$$cohesion(c_j; cat) = \sum_{t \in c_j} MI(t, cat)$$

MI measures the amount of information each term t tells us about category cat: $MI(t, cat) = p^{'}(t|cat)p(cat)log(\frac{p(t|cat)}{p(t)})$. $p(t|cat)$ is the probability that term t appears in comments in cat. $p^{'}(t|cat)$ is a correction to $p(t|cat)$ that gives every term a non-zero probability of occurrence across all categories. Therefore we have $p^{'}(t|cat) = \alpha p(t|cat) + (1-\alpha)p(t)$ as a smoothed probability that a comment contains term t given that it belongs to category cat. α is between 0 and 1. In practice we select a smoothing factor of $\alpha = 0.9$. $p(t)$ is the fraction of all comments containing t; and $p(cat)$ is the fraction of comments belonging to category cat. To prevent comments with more terms from receiving higher cohesion values, we also considered a version that divides cohesion by the number of terms in c_j. Experimentally, we find that this normalized version yields qualitatively similar results.

$$p(t|cat) = \frac{count(cat, t)}{count(cat)}, p(cat) = \frac{count(cat)}{n} \tag{3.4}$$

The next set of content-based features rely on NLP-style analysis of the comments:

- *Readability*: We measure the readability of a comment by its SMOG score [64], which estimates the years of education needed to understand a piece of writing. SMOG considers the number of words with more than three syllables (poly Syllables) and the number of sentences in the text.

$$SMOG = \sqrt{polySyllables * 30.0/sentences}$$

- *Subjectivity vs. objectivity*: We measure the subjectivity/objectivity of each comment using the open source NLP tool LingPipe [10].

- *Verb+Noun count*: A simple count of verbs and nouns.

The last set of content-based features compare the comment text to the article the comment is attached to:

- *Comment-article overlap*: This feature measures the overlap between terms in the article abstract and the comment.

- *Comment-article polarity*: Finally, we measure if the polarity of each comment (positive or negative) matches the polarity of the article (using LingPipe [10]): 1 for agreement; 0 for disagreement.

3.6 Experiments

In this section, we evaluate the quality of the community-preference prediction model using the features described in the previous sections.

3.6.1 Data

For our dataset, we crawled the most-Dugg stories of the past 365 days in November 2008, resulting in a corpus of 9,000 Digg stories containing 247,004 comments submitted by 47,084 unique contributors. Each story belongs to one of eight major categories: Technology, World and Business, Science, Lifestyle, Entertainment, Sports, Gaming, and Offbeat. We focused our collection on these older pages since the commenting and rating activity has most likely stabilized for these stories, leading to a more reliable analysis of the comments.

3.6.2 Evaluation Method

Our evaluation is designed with three goals in mind. First, we aim to compare the learning-based ranking approach versus alternative approaches, to understand if the model does indeed capture salient features for predicting community preference. Second, we isolate the features used by the model to gain a better understanding of which comment features are good predictors of community preference. Finally, we explore an extension to the model for identifying and promoting high-quality comments that may have been overlooked.

As a baseline, we can measure the effectiveness of the learned model by comparing the predicted rank order of the comments to the ground truth rank order, as determined by the ground truth comment community ratings. Recall that it is important that the predicted comment rankings be of especially high-quality for the top-k comments for small k, since users and applications are typically most interested in these high-quality comments. Errors in ranking prediction at lower ranks are of less importance (e.g., swapping the 200th and the 201st comment). Hence, we evaluate the quality of the predictions using the well-known Normalized Discounted Cumulative Gain (NDCG) measure for evaluating the quality of top-k lists [16]. NDCG reflects this intuition by reducing the penalty of ranking errors logarithmically in proportion to the position of the comment. Formally, the discounted cumulative gain (DCG) is found for a top-k list as:

$$DCG_k = \sum_{i=1}^{k} \frac{fav_i}{\log_2(1+i)}$$

where fav_i is a favorability score for the comment at position i. We define the favorability score as its rank complement: $fav_i = N - Rank_i + 1$. For comparison across top-k lists for different articles, DCG is normalized by the *ideal* discounted cumulative gain at k. The ideal DCG ($iDCG_k$) is found by sorting the comments in

order of their comment community rating and calculating DCG as above, resulting in $NDCG_k$:

$$NDCG_k = \frac{DCG_k}{iDCG_k}$$

NDCG ranges from 0 to 1, with higher-scores indicating greater agreement between the predicted rank order and the ideal rank order (based on the comment community ratings).

In all of the experiments reported here, we train and test the model using 10-fold cross validation and a 20-80 train-test split. After randomly sampling 24,000 comments from the dataset, the data is randomly split into 10 parts. We train the model over two of the parts (including the ground truth comment community rating) and then test the model over the remaining eight parts (for which the model has no access to the ground truth comment community rating). This procedure is repeated 10 times; the results are averaged over the 10-folds.

3.6.3 Model Comparison

First, we compare the proposed model – denoted here as the Social Web Comment Prediction *SWCP* model – against two alternatives: a random ranking model and a time-of-posting based ranking model. In the random ranking model, comment order is purely random. This simplistic model provides us with a baseline against which to compare the developed models. The second model is a time-of-posting ranking model. Recall that in Figure 3.9, we saw how comment posting time has a strong impact on its community rating, since earlier comments have greater visibility in the community. It might be reasonable to conjecture that posting time is all that matters. Concretely, this model assigns rank order to comments based solely on time-of-posting, i.e., comments arriving in the order $\{c_1, c_2, ..., c_n\}$ are ranked $\{1, 2, ..., n\}$.

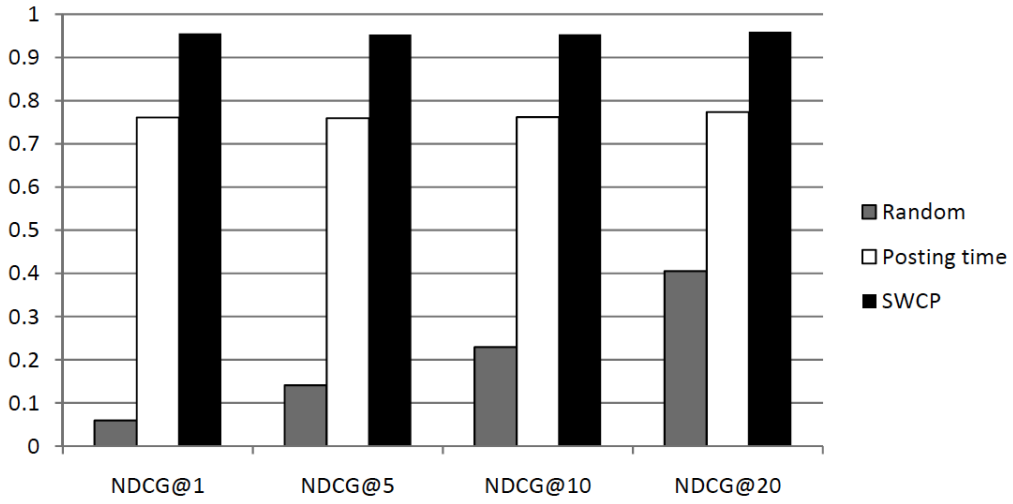

Figure 3.10: Comparing the SWCP model versus alternatives.

Figure 3.10 shows the performance of the three models across four different NDCG k-values: NDCG@1, NDCG@5, NDCG@10, and NDCG@20. First note that both the comment-posting time model and the SWCP model outperform the random model for all NDCG metrics. Second, although the comment-posting time performs reasonably well, it alone is an insufficient determiner of comment community preference. We see that the inclusion of the user-based and comment-based features results in around a 25% improvement across all NDCG metrics. What is especially encouraging is that the model performs extremely well for the top-1 comment, meaning the model almost always correctly identifies the top-1 comment regardless of its posting time. The similarly good results for 5, 10, and 20 are also encouraging, validating the premise that comments, although a "messier" form of user-based annotation (compared to tags and ratings), do contain implicit quality signals that can be mined and used for automatic comment extraction by community preference. This has strong positive implications for the success of new comment-based applications (e.g., enhanced information organization, summarization, content retrieval, and visualization), as well as the continued success of the Social Web in the presence of growing

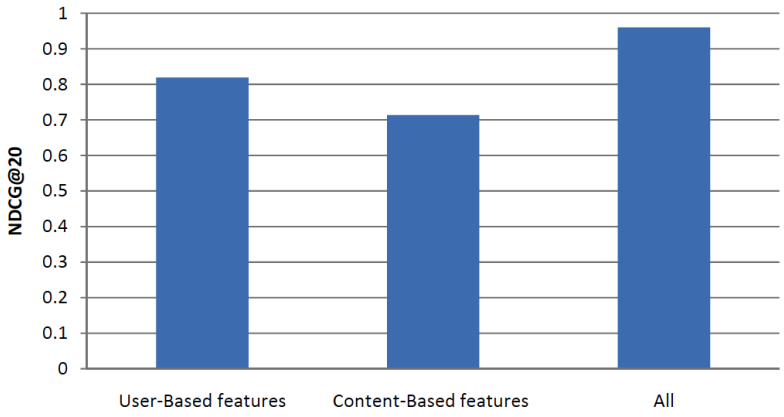

Figure 3.11: Comparing feature sets.

spam and low-quality comments.

3.6.4 Feature Study

Given the performance of the Social Web Comment Prediction model, what impact do the user-based and content-based features have on the prediction quality? Since evaluation of all possible feature combinations would be computationally expensive, we isolate the features in groups to better understand which features are good predictors.

First, we train two models – one using only user-based features (recall Section 3.5.2) and one using only content-based features (recall Section 3.5.3). Figure 3.11 shows the performance of the user-based model, the content-based model, and the full feature model for NDCG@20. We find qualitatively similar results for other values of NDCG@k (k=1, 5, 10). The user-based features alone do a better job than content-based feature alone, however, both approaches perform significantly less well than the full combination of features. We view the user-based features as a "prior" on the preference of the community for the user's comments. Only in combination with the actual comment text can we predict the community preference with

good success. This negates the hypothesis that power users wield excessive control over comments (unlike the article promotion feature of Digg, which many presume is heavily influenced by power users).

To better understand the relative impact of particular user-based and content-based features, we next train and evaluate six models – one for each of the three user-based feature groups, and one for each of the three content-based feature groups. Table 3.2 reports the NDCG@k for k=1, 5, 10, and 20 for each of these six feature groupings.

Feature group	**NDCG@1**	**@5**	**@10**	**@20**
User activity and interest	0.61	0.62	0.65	0.70
User popularity	0.64	0.65	0.67	0.72
User comment history	0.66	0.69	0.71	0.73
Content statistics	0.62	0.65	0.67	0.71
Content NLP features	0.64	0.67	0.68	0.72
Comment-article	0.66	0.68	0.70	0.73

Table 3.2: Evaluation of different feature sets

For the user-based feature, the user comment history feature group (recall that this includes the history of a user's previous comment ratings and the number of replies those comments have received) shows the strongest impact. This indicates that some users have a specialty for writing comments that are appreciated by the community; again, we can interpret this feature as a "prior" on a given comment's quality. Also note that content-based features are important; two of the top-three feature groups are content-based. We find it interesting that user activity and interest level – based on articles submitted, length of community membership, and category activity – is the single weakest performing feature group. Authoring comments that are perceived as high-quality by the community is largely independent of the user's activity level. Our hypothesis is that there are fundamentally different user types

within a Social Web community: article submitters, article raters, commenters, etc. Exploring these different user types and their inter-relationship is an area deserving of further study.

In the final feature study, we explore the importance of content-based features for appropriately modeling the domain. We begin by assuming that our model has access to all user-based features. Could it be that comments are not really "messy"? And that by adding a single content-based feature we can equal the performance of the full feature model? Intuitively, this would mean that the comments contain some clear quality indicators once we factor in the "prior" for the user contributing the comment.

Feature group	**NDCG@1**	**@5**	**@10**	**@20**
All user-based features (A)	0.74	0.74	0.75	0.81
A + Text length	0.76	0.76	0.77	0.83
A + Upper case	0.74	0.74	0.75	0.81
A + Entropy	0.73	0.74	0.75	0.81
A + Informativeness	0.73	0.74	0.75	0.82
All features (user+content)	0.94	0.95	0.95	0.96

Table 3.3: Evaluation of combination of all user-based features with a single content based feature

Table 3.3 reports the NDCG values for the baseline model considering only user-based features, plus four models that consider the baseline plus a single content-based feature only (text length, upper case, entropy, informativeness). In all, however, the content-based features are quite valuable. This indicates that comment content is complex, and that the community's preference for a comment is not driven by a simple feature. Instead, we see the need for full content analysis to capture this complexity.

3.6.5 Rank Boosting

As we have seen in Figure 3.9, the comment posting time has a strong influence on the visibility of a comment and the resulting comment community rating. In this last experiment, we are interested in further exploring this phenomenon, as a first step toward breaking the rich-get-richer visibility cycle. As an example, consider the four comments A, B, C, and D and their actual comment community ratings as illustrated in Figure 3.12. Applying a simple comment posting time ranking to these comments results in the rank order $\{A, B, C, D\}$. After applying the Social Web Comment Prediction model, we would ideally find the rank order $\{A, D, B, C\}$. This rank order is in strict order of the community ratings. Indeed, we have seen how the proposed model performs well on this problem. It might be reasonable, however, to claim that comment D is the most preferred comment. Based on its late arrival time, but high community rating, we could assert that comment D has been most appreciated by the community relative to its smaller community visibility. This intuition motivates this last exploratory experiment.

Referring back to Figure 3.9, we propose to re-scale the comment community ratings for each training instance with respect to the average community rating for other comments posted in the same order position. In this way, we can evaluate a post arriving 4th (as in the example with comment D) against *all other comments* in our training data arriving 4th. The intuition is the further a comment's rating is from the average relative to other comments in the same position, then the more the comment's rating should be rewarded or punished.

Concretely, for a comment in the j'th position attached to a Social Web object i, we can define the *boosted* comment community rating $\hat{r}_{c_{ij}}$ with respect to all k comments at this same position as:

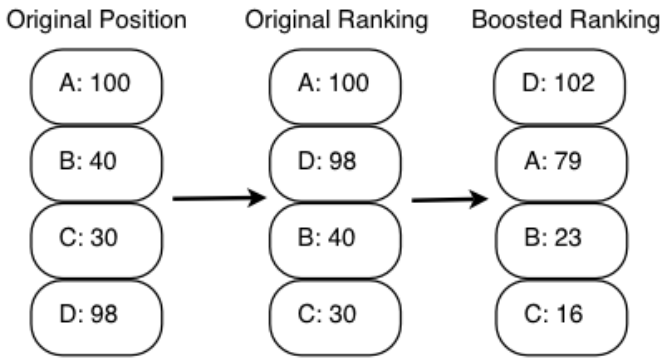

Figure 3.12: Example illustrating the original time-of-posting position for each comment, the predicted ranking according to the SWCP model, and the boosted ranking using the positional boost modification.

$$\hat{r}_{c_{ij}} = r_{c_{ij}} + r_{c_{ij}} \times \frac{r_{c_{ij}} - \bar{r}_{c_j}}{\sqrt{\frac{1}{k}\sum_{i=1}^{k}\left(r_{c_{ij}} - \bar{r}_{c_j}\right)^2}}$$

where $\bar{r}_{c_j}$ is the mean comment score at position j ($\bar{r}_{c_{ij}} = \frac{1}{k}\sum_{i=1}^{k} r_{c_{ij}}$) and the denominator is the variance of these comment scores. So a comment with a large rating in a position with a small average rating and small variance would be promoted to a new boosted rating. Returning to our example, suppose the (average, variance) pairs of all comments at positions 1 to 4 are: (148, 235), (119, 193), (105, 169), and (91, 158). Applying the boosting formula results in the rank order $\{D, A, B, C\}$. Since comment D's original rating is much higher than the average rating for other comments at the same position, it is boosted from a score of 98 to 102. More importantly, comment A underperforms for its position and is penalized from 100 to 79.

3.7 Conclusion

We have proposed and evaluated a regression-based learning model for automatically identifying comment quality within a Social Web community based on the

community's preferences. We examined the impact of different comment features like visibility, user reputation of the comment's author, and the content of the comment itself to understand the influence of these features on the overall community's preference for comments. Through experiments, we find that the proposed approach results in significant improvement in ranking quality versus alternative approaches. Additionally, we study an extension to the model for balancing the visibility of a comment with its intrinsic quality.

4. LABELING TWEETS: A SEMANTIC-GRAPH APPROACH*

4.1 Introduction

In this chapter we consider the problem of labeling short text data that is generated by the web community. Examples include the tweets in Twitter micro-blogging website. Labeling will be a method of categorization of the information to make it easier for the users to find relevant content. It is crucial to understand the context of the short text in order to recommend appropriate labels.

The self-curation of the real-time web could be achieved via *user-driven linking*, in which users annotate their own status updates with lightweight semantic annotations – or *hashtags*. On Twitter, for example, these hashtags are inserted into tweets by users and serve many functions. For example, some reflect categorical information about the tweet as in Figure 4.1, where both have been annotated with the hashtag #health. Some hashtags reflect events related to a tweet (e.g., #ht2012) and many others reflect the sentiment of the tweet (e.g., #Iloveapple, #sucks). And of course, as user-generated descriptors, some are nonsensical or of interest only to the user posting the hashtag.

By linking status updates to hashtag-like semantic descriptors, users provide a potentially scalable mechanism to organize the real-time web as it continues to grow. As users continue to post status updates with hashtags, there will always be additional semantic cues for organizing these updates. For example, as new issues become associated with the "Health" concept, we would expect to see new updates using the #health hashtag. In this way, the user-driven semantic annotation of the

*Reprinted with permission from "Predicting semantic annotations on the real-time web" by Elham Khabiri, James Caverlee and Krishna Y. Kamath, 2012. In *HT*, 219-228, Copyright 2012 by Association for Computing Machinery.

Figure 4.1: Two sample tweets annotated with the hashtag #health.

real-time web could provide an evolving framework for improving information navigation in these systems (by linking similar updates according to common hashtags), by inducing concept hierarchies over these status updates (so that #cancer-related updates are organized under the umbrella of #health), for supporting serendipitous exploration of the real-time web, improving the recall of search operators (by returning both #apple and #mac related updates for queries about the company), and so on. Indeed, a recent study of Twitter search shows that hashtags are popular as queries, and that these queries are often repeated so that users may monitor search results [90]. By linking untagged updates with hashtag-like semantic descriptors, such searches could have expanded coverage.

Generally assignment of appropriate annotations results in a more accurate information retrieval for social searches occur in micro-blogging space. The reason behind why people would use twitter social search over traditional web search is to find temporally related information, crowd sentiment about a content and information about other users' interest [90]. Unfortunately, there is evidence that hashtag growth is not keeping pace with the growth of the overall real-time web. In a random sample of 3 million tweets, we find that only 10.2% contain at least one hashtag, meaning that 89.8% are un-labeled and would be left out of any hashtag-oriented search or

monitoring application. In addition, there is mounting evidence that many hashtags may convey little semantic information or are being used as tools of spammers and other polluters of these systems [38, 44, 51]. Hence, in this chapter we explore the possibility of predicting hashtags for un-annotated status updates. Can we determine the appropriate semantic label for an update?

Toward this end, we propose and evaluate a graph-based prediction framework in which terms in status updates are linked to hashtags based on their co-occurrence. Since many relevant hashtags may not co-occur with all possible terms, we develop a path aggregation technique for scoring the closeness of terms and hashtags in the graph. In this way, high-value hashtags may be associated with status updates, even if no terms in the update have ever co-occurred with the hashtag. Additionally, we augment the baseline method with a pivot term selection approach for identifying high value terms in status updates, and a dynamic sliding window for recommending hashtags reflecting the current status of the real-time web. Experimentally we find encouraging results in comparison with Bayesian and data mining-based approaches.

4.2 Predicting Semantic Annotations

In this section, we formalize the problem of predicting semantic annotations for the real-time web and introduce a hashtag graph-based prediction framework.

4.2.1 Problem Statement

Let $T = \{T_1, T_2, ..., T_n\}$ be the set of status updates (i.e., tweets), and $T_i = \{u_1, u_2, ..., u_m\}$ be a set of unigram terms, and $H = \{h_1, h_2, ..., h_m\}$ be the set of hashtags. Our goal is for an unlabeled status update T_i to predict a hashtag h_j that "correctly" annotates the update. Of course, it is challenging to determine what is the "correct" choice of hashtag. In one direction, the evaluation of hashtag prediction can be based on a user study in which human subjects are asked to evaluate the

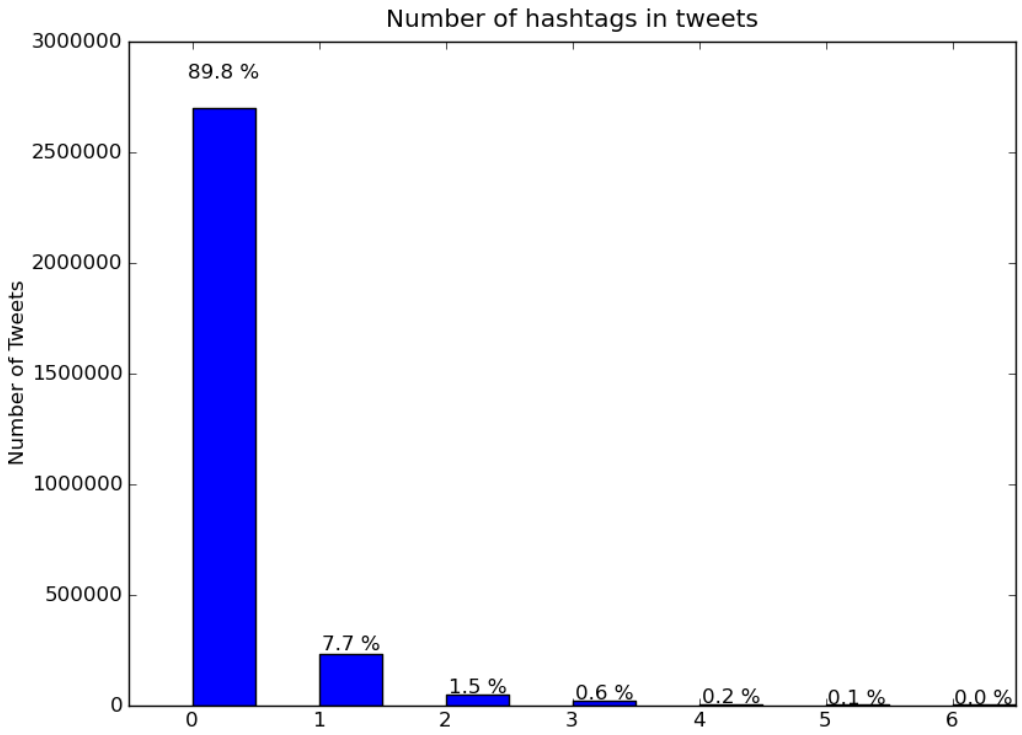

Figure 4.2: Most tweets are annotated with no hashtags. In a random sample of 3 million tweets, we find that 7.7% contain exactly one hashtag, and 2.5% contain more than one hashtag.

quality of predicted hashtags for each of the testing tweets. A recent study [23] argues that human evaluation of tags may lead to errors in assessment due to multi-lingual tags, missing context, differences of level of details, and the interdependence of tags. Alternatively, we can adopt a purely machine-based evaluation framework in which the prediction model is built over a training set and then used to predict the hashtags for a test set. In this case, the hashtags themselves are removed from the test set and then the quality of the prediction is in identifying the actual hashtag that had been used. Such an approach, while providing less flexibility (e.g., by not accepting #nba as a reasonable tag for a sports-related tweet actually annotated with #basketball), does provide for fast evaluation and comparison across multiple methods. Hence, we adopt this second approach.

Concretely, we adopt an evaluation framework in which a portion of the data is used as a training set for learning the prediction model, and a separate testing set is used for evaluation. The model is used to predict the hashtags of test tweets in which all the hashtags are removed. The predicted k tags are denoted t_{pred}. The actual tags applied to the tweet are denoted t_{real}. For varying values of k, we can evaluate the quality of hashtag prediction using precision:

$$prec = \frac{|t_{real} \cap t_{pred}|}{|t_{pred}|}$$

where predicting only hashtags that are actually used results in a precision of 1, whereas predicting none of the correct tags actually used results in a precision of 0. We additionally evaluate the quality of hashtag prediction using recall:

$$rec = \frac{|t_{real} \cap t_{pred}|}{|t_{real}|}$$

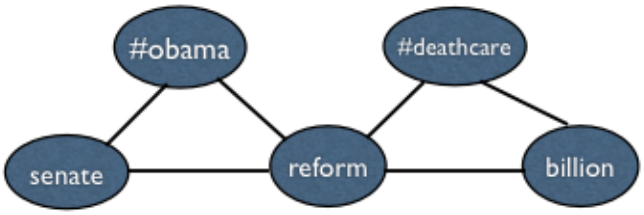

Figure 4.3: Although "senate" and "#deathcare" have not appeared together in any tweets, the two are related, as revealed by the short path (2 hops) in the semantic graph.

where identifying all of the correct hashtags results in a recall of 1. Finally, we also consider the combined F-measure:

$$f = \frac{2 \times prec \times rec}{prec + rec}$$

We measure the overall precision, recall and F-measure by averaging over all testing tweets.

4.2.2 Hashtag Graph-Based Prediction

Given the overall goal, we propose in this section a graph-based prediction approach. The core idea is to identify implicit relationships among the hashtags and terms used in tweets to build a semantic graph that may then be used to connect the terms in unlabeled tweets to the appropriate hashtags. The baseline assumption is that terms and hashtags that are used together are related and hence close in terms of meaning. For example, Figure 4.3 shows a subgraph built over a large Twitter dataset (described more fully in the experimental evaluation) in which a term like "senate" is linked to "reform" due to the use of both terms in many tweets. Similarly, "senate" and the hashtag "#obama" are linked due to their co-occurrence. However, strictly considering co-occurrence alone will miss the implicit connection between "senate" and "#deathcare". Returning to Figure 4.1, we find that terms

like "sick" and "patient" are close in the semantic graph to the hashtag "#health". By identifying these implicit connections across all of the terms used in an unlabeled tweet, the proposed approach seeks to find hashtags that are close in terms of this semantic graph. Hence, for a tweet T, we can estimate the appropriateness of a hashtag h as an aggregation operation over all of the terms occurring in T:

$$score(T, h) = \sum_{t_i \in T} p\text{-}score(t_i, h) \tag{4.1}$$

where $p\text{-}score(t_i, h)$ is an estimate of $p(h|t_i)$ – the conditional probability of the hashtag being used, when t_i is observed.

However, naive application of such an approach will face several challenges. First, how should evidence from different terms from a single tweet be aggregated to find the consensus of the tweet? In other words, a tweet containing terms like "senate" and "healthcare" may be closely linked to many candidate hashtags. In what ways can we distill the most likely hashtags from a long list of candidates? Second, aggregating the evidence across all terms in a tweet may lead to topic drift, in which particular terms are closely linked to hashtags that are not at all relevant to the overall tweet. For example, the term "state"' in the first tweet shown in Figure 4.1 may be linked to hashtags associated with mental states, states like Texas and Oregon, and other concepts not at all linked to the hashtag "#health". Third, the probability of a hashtag given a term may change over time. For example, the term "obama" will be closely linked with different terms and different hashtags based on the political debate of the day, whether the election is upcoming, and so on. Hence, careful determination of the temporal relationships between terms and hashtags is important.

With these challenges in mind, we now detail three specific steps toward hashtag graph-based prediction: (i) a path aggregation technique for scoring the closeness

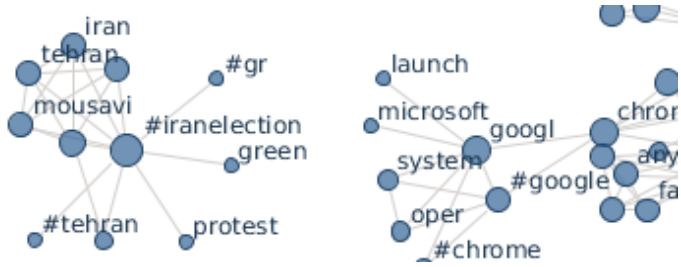

Figure 4.4: Relationship among hashtag and terms. The left side shows terms and hashtags related to the Iran election; the right side is technology-centric.

of terms and hashtags in the graph; (ii) pivot term selection, for identifying high value terms in status updates; and (iii) a dynamic sliding window for recommending hashtags reflecting the current status of the real-time web.

4.2.2.1 Linking Terms and Hashtags

First, we build a semantic graph and propose a path aggregation technique for scoring the closeness of terms and hashtags in the graph. We build a graph $G = (N, E)$ with nodes $N = \{n_1, n_2, ..., n_m\}$ in which n_i is either a term or a hashtag and edges $E = \{e_1, e_2, ..., e_r\}$ in which e_j is the weighted edge between two nodes. To avoid noise and to keep our graph less polluted we only create an edge between two nodes when the number of co-occurrences is greater than a threshold. The co-occurrence is measured by considering all tweets in the training set and counting the number of times two elements (either terms or hashtags) occur together in the same tweet. In this way, we may filter out non-important edges that have happened by chance. A sample graph is illustrated in Figure 4.4, which shows the relationship between terms and hashtags.

But what is the appropriate weighting function for edges between nodes? This weighting function can be used to identify the relative "closeness" of terms and hashtags that are directly connected. In one direction, the co-occurrence count itself

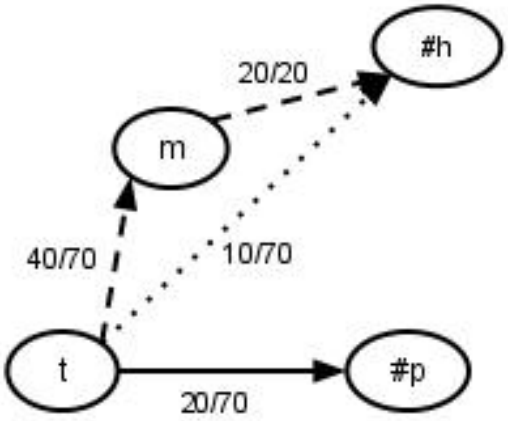

Figure 4.5: The score of hash #h related to term t is calculated by the summation of all the path scores between these two nodes.

may be used. Consider the case that terms A and B have co-occurred 10 times together, and both A and B occur across all tweets exactly 10 times each. So, A and B always co-occur together and never apart. Now suppose A appears 100 times, but only in 10 cases did it co-occur with B. In this case, the "closeness" of A and B is less than in the first case. Hence, we normalize the co-occurrence value by the number of the times a term has appeared in the whole corpus which is equal to the number of outlinks of that node. This *normalized weight score*, $NW_{(n,n+1)}$, is the normalized weight of the edge between node n and node $n+1$:

$$NW_{(n,n+1)} = \frac{W_{(n,n+1)}}{\sum_{p \in Outlink(n)} W(n,p)} \tag{4.2}$$

where $W_{(n,n+1)}$ is the co-occurrence count of the two elements n and $n+1$ (terms or hashtags). By this normalization we consider the amount of the node devotion to the relationship with another node. Therefore an edge to a more general term will receive smaller weight in comparison with an edge to a more specific term. Now the question is how to measure the "closeness" of nodes that are more than one hop

away? What will happen if the relevant hashtag was found two or three hops away? Shall we penalize the score of the hashtags that are located farther from a particular term? And to what degree?

To formalize the problem we say that the hashtag h is reachable from term t_i in radius m as $t_i \rightsquigarrow^m h$. For a single path from a term to a hashtag, we propose to consider the product of all the edge weights in the path, where the edge weights themselves are decayed by a factor β. The decay factor is to penalize the nodes that are far from the source node, so that we still consider them as candidate hashtags but with lower significance than ones that are directly connected in the graph. The score for a path is then:

$$score\_path(t_i \rightsquigarrow^m h) = \prod_{n=0}^{m-1} NW_{(n,n+1)} * \beta^n \tag{4.3}$$

where the normalized edge weights between nodes are decayed, and so the farther a hashtag is from a term source node, the less score it gets.

To find the overall score of a hashtag h from term t_i we measure aggregation scores of all of the paths existing between them. So that if a hashtag is reachable by more than one path it shows more relevance to the term in comparison with the case that it is only reachable by one path. Hence, this aggregated path score is:

$$p\text{-}score(t_i, h) = \sum_{m=1}^{M} score\text{-}path(t_i \rightsquigarrow^m h) \tag{4.4}$$

where we consider all paths from a term to a hashtag. For example in Figure 4.5 hashtag h is reachable from a term t once in 1 hop (the dotted path,) and the other time through 2 hops (the dashed path). In this way, we link terms to hashtags.

4.2.2.2 Selecting Pivot Terms

Given the semantic graph and the method for linking terms to hashtags, the aggregation method described in Equation 4.1 can be applied immediately. However, by considering all terms in a tweet for finding appropriate hashtags may introduce noise in the case of spurious term-hashtag connections caused by considering isolated linkages between terms and hashtags without considering the overall tweet content. For example for the tweet "So there is actually a python module called pyjamas", many of the terms are not significant for predicting an appropriate semantic annotation; "so", "there", "is", and so on are relatively common terms and they convey little information about the tweet. In contrast, "python", "module", and "pyjamas" are all strong cues.

Hence, we propose to select a subset of terms from each tweet based on their high information content. This *pivot term selection* results in keeping the model small and eliminating terms that are ineffective for tag prediction. While there are a number of ways to select pivot terms, we consider two approaches – by inverse document frequency and by entropy.

To select pivot terms by inverse document frequency measure (IDF), we consider the number of times a term was used in all the tweets – df_t – within the training set.

$$IDF(t) = log\frac{N}{df_t} \tag{4.5}$$

where N is the total number of tweets in the training set. Hence we identify the terms with high IDF and eliminate the more general terms with low value.

For entropy-based pivot term selection, we identify terms with low entropy (which tend to be more specific) and eliminate terms with high entropy (which tend to be

more general non-informative terms). The entropy of term t is measured as:

$$Entropy(t) = -\sum_{h_i \in H} p(h_i|t) \times log(p(h_i|t)) \tag{4.6}$$

where H is the set of all hashtags that co-occur with term t. By selecting as pivot terms those terms that are low in entropy, the goal is to find good predictors of hashtags. To illustrate, Table 4.1 shows a sample of terms with high entropy versus those with low entropy in a large collection of tweets (described in the following section). The terms with lower entropy are more specific and terms with higher entropy are more general terms.

Low Entropy	High Entropy
twade sherlock	win house
vancouver perception	save chance
tweekly intriguing	post prize
wesson legend	good top
naraku equivalent	american person
crunchy irm	end night
tempting tub	tip group
jumper drinking	stop hot
chilli whistle	week nice

Table 4.1: Sample of terms with high/low entropy.

4.2.2.3 Sliding Windows

Finally, since the real-time web is constantly changing, we augment the baseline hashtag prediction approach with a *sliding window*. The intuition is that the recency of hashtags is a strong indicator of their appropriateness for annotating tweets. For example, events such as Gaddafi's death or the Super Bowl, shifting user interests, or announcements of new products will drive a changing portfolio of hashtags in use by users of the real-time web. Thus, a higher importance can be assigned to more recent hashtags than those introduced a long time ago.

Concretely, we propose to build the semantic graph based on the past Δ time,

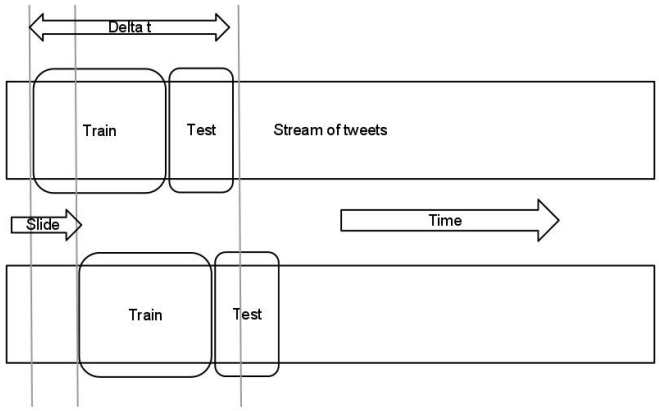

Figure 4.6: From the stream of tweets we construct a time-window of Δ and split the data into 80-20 train-test sets within each window. We repeat the experiments for all sliding windows.

rather than considering the entire history. The sliding window could be an hour, day, week, or month. In this way, the predicted hashtags can be based on this sliding window as illustrated in Figure 4.6, reflecting the current composition of hashtags.

Additionally, the model may be smoothed by considering a mixture of both the most recent window and the global history. For example if we build the model based on the past day, could we improve it by considering the information from the past week? Therefore we suggest a smoothing process that takes into account both the recent history and the complete history:

$$\textit{smooth-score}(t_i, h) = 0.9 * \textit{p-score}(t_i, h) + 0.1 * G(h|t_i) \tag{4.7}$$

where $G(h|t_i)$ is the global probability of hashtag h given term t_i. Also note that the global model can be calculated offline, so that it can be efficiently incorporated into the sliding window approach.

4.3 Evaluation

In this section, we present an experimental study of hashtag recommendation. We describe the dataset, metrics used, introduce several alternative adaptations of tag prediction in social media to the problem of hashtag prediction, and present the results of a comparative study.

4.3.1 Dataset

For the experimental evaluation, we adopt the Stanford Twitter dataset containing 344 million Twitter posts from 20 million users covering a 6 month period from 06/01/2009 to 11/31/2009 [97]. This dataset contains about 20-30% of all public tweets published on Twitter during this time frame. After removing tweets with empty text, we arrive at a dataset described in Table 4.2. Eliminating terms and hashtags with length less than 2 and those that were used fewer than 10 times in tweets, we arrive at nearly 500K unique terms and 100K unique hashtags in the dataset. We randomly split the data into an 80/20 mix, so that 80% of the tweets with hashtags are used as training and 20% of tweets with hashtags are for testing.

Total number of tweets	344,139,347
Total tweets with hash	36,558,421
Size of term dictionary	502,684
Size of hash dictionary	134,522

Table 4.2: Statistics of Twitter dataset.

4.3.2 Alternative Methods

As we discussed in the related work section, there have been a number of studies of tag recommendation over traditional social media. We now describe adaptations of several of these alternative approaches, in which we customize the techniques for

hashtag prediction.

4.3.2.1 Adapting Flickr-Based Tag Recommendation

The first approach was developed in the context of tag recommendation for photos on Flickr [85]. In this context, it is assumed that each photo has already been tagged by some set of users. Based on the co-occurrences of these tags with tags associated with other photos, the method can recommend additional tags for the baseline photo. The authors propose two aggregation methods for scoring and ranking candidate tags in order of their appropriateness – by voting and by summation. The *voting-based method* considers the number of times that a candidate tag was seen. As an example consider A and B as the two input tags for a photo. Suppose A co-occurs with $\{M, N\}$ and B co-occurs with $\{M, P\}$. The votes will be $\{M : 2, N : 1, P : 1\}$, meaning that M will be the most highly-rated new tag to be recommended. The *summation-based method* additionally uses the co-occurrence value of the tags. Suppose for the same example that the co-occurrence values are: $A \rightarrow \{M : 1, N : 9\}$ and $B \rightarrow \{M : 2, P : 10\}$. Then the summation-based method will score the three tags as: $\{M : 3, N : 10, P : 10\}$, where now M is the lowest-score tag. Translating from Flickr tag recommendation to our context, we can consider each term as an object and then consider all of the hashtags that were used with this term across all tweets. Therefore we have each tweet made of p terms: $T_i = \{t_1, t_2, ..., t_p\}$. Hence, the voting-based method becomes:

$$vote(h, T) = \sum_{t_i \in T} vote(h, t_i)$$

where

$$vote(h, t_i) = \begin{cases} 1 & \text{if } h, t_i \text{ co-occur} \\ 0 & \text{otherwise} \end{cases}$$

in which we consider all of the hashtags that have co-occurred with each of the terms in tweet T and count the number of times a hash has co-occurred with each of the t_i in T. For the summation-based method we can consider the number of co-occurrences of hashtag h and the terms in a tweet T.

$$sum(h, T) = \sum_{t_i \in T} count(h, t_i)$$

where $count(h, t_i)$ denotes the number of co-occurrences of hashtag h and the terms in a tweet T. In both methods the scores are additionally normalized with a promotion score in which the stability of the term, descriptiveness of hashtag and the rank of hashtag in the co-occurrence list of the terms is considered. For more explanation we refer the interested reader to [85].

4.3.2.2 Bayesian Prediction

The second approach is based on Bayesian principles and also originates in image-based tag prediction [94]. Adapting this method, we can consider the co-occurrence of hashtags and terms along with the user tag history. Here, the probability of suggesting a hashtag h to a user u for the resource t_i is defined as:

$$p(h|u, t_i) = \frac{p(u, t_i|h) * p(h)}{p(u, t_i)} = \frac{p(u|h) * p(t_i|h) * p(h)}{p(u, t_i)} \tag{4.8}$$

where $p(h|u, t_i)$ is the probability that user u uses hashtag h to annotate resource t_i, $p(u, t_i|h)$ is the posterior probability of user u and resource t_i given a hashtag h, and $p(h)$ is the prior probability of hashtag h. Having the score of a hashtag h for each of the terms t_i we can find the total score of a hashtag for the whole tweet T as:

$$p(h|u, T) = \sum_{t_i \in T} p(h|u, t_i) \tag{4.9}$$

In this method since the user tagging history is taken into account, the score measured for each hashtag is a personalized score.

4.3.2.3 Association Rule Mining

The third approach is based on market-basket data mining principles for predicting tags [29][93]. In the market-basket model we have a large set of items and a large set of baskets containing a subset of items [3]. We are interested to identify the items that are purchased together frequently in a basket. This model generates the association rule of the form $\{I_1, I_2, .., I_n\} \Rightarrow \{h\}$ meaning that finding $I = \{I_1, I_2, .., I_n\}$ in a basket, there is a good chance of finding h in it. In particular, the popular association rule mining approach can be used to identify interesting relationships among terms and hashtags based on the probability of occurrence of the terms with their related hashtags. Adapting the market-basket model to the hashtag prediction problem, the baskets are the tweets, and the items are the terms and hashtags appearing in a tweet. The goal is to find the most probable hashtags when a set of terms I has been observed in a tweet. In this model we care about the term-hashtag pairs that appear frequently together and are considered to have high *support*. Another metric called $confidence$ implies the probability of finding h knowing that I has occurred. The rules with high confidence and support construct the useful association rules. Here we define $supp(I)$ as the number of tweets in which I has appeared and $conf(I \Rightarrow h)$ as the probability of using hashtag h when I is observed in a tweet as a set of terms:

$$conf(I \Rightarrow h) = P(h|I) = \frac{supp(I, h)}{supp(I)} \tag{4.10}$$

which is the number of times the terms I and hashtag h appear together divided by the number of times that the terms I appeared in the training dataset. In this way, association rules are used to find interesting term-hashtag relationships. The length

of association rule can vary. In practice, the most interesting rules have a length of less than 3 for short text dataset. Hence, we first extract all possible association rules from the training set, keep only those of length 3 or less. To predict the hashtags for a new tweet, the rules with the same input terms and high confidence and support are used. As an example for the tweet "Freedom for journalism in Iran", we can consider the rules with support more than a threshold (30 in this case), resulting in the following high confidence rules: $\{freedom, iran\} \rightarrow \#iran$, $\{iran, journal\} \rightarrow \#iranelection$. Therefore the suggested hashtags will be $\{\#iran, \#iranelection\}$.

4.3.3 Experimental Results

We now evaluate the performance of the proposed graph-based approach to predict annotations. To do this we use the metrics described in Section 4.2.1 and the Twitter dataset described in Section 4.3.1. In particular, we perform three set of experiments: (i) to estimate the parameters and pivot selection methods used in the graph based approach; (ii) to compare the performance of our approach with the alternate approaches described earlier in this section; and (iii) to analyze the graph-based prediction approach.

4.3.3.1 Parameter Estimation

We estimate three parameters used by the graph-based approach: (i) the number of hops to consider from a pivot term; (ii) the decay factor (β) for penalizing nodes far from the pivot term; and (iii) the length of the sliding window (Δ). In addition to these parameters, we also compare the two pivot term selection methods – by entropy and by inverse document frequency.

Number of Hops: In this experiment we estimate the maximum number of hops to take from a pivot term to determine candidate hashtags for annotation. For example, while a value of one hop will consider only the immediate neighbors of a term, a choice

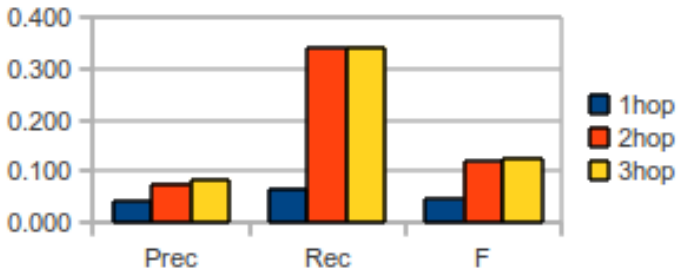

Figure 4.7: Increasing the number of hops identifies more relevant hashtags in the semantic graph.

of hops greater than 1 will consider hashtags that do not directly co-occur with the pivot term but are related to it. Hence in this experiment, we tried different number of hops after setting $\beta = 0.80$ and $\Delta = 1$ week. The result of this experiment is shown in Figure 4.7. We observe a large improvement in recall as the number of hops increases to 2, suggesting that these nearby hashtags are good candidates (even if they have not co-occurred directly with the terms in a particular status update). We also note that the recall and F-measure are nearly the same comparing hops 2 and 3, meaning that additional exploration of the semantic graph identifies few additional significant hashtags. Since this larger exploration comes at a larger computational cost, we set the number of hops to 2 for the remainder of the experiments.

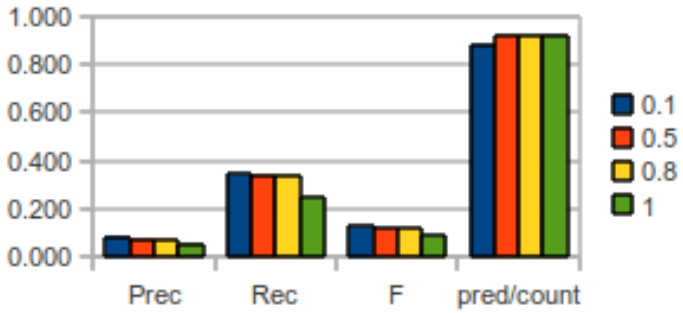

Figure 4.8: A smaller decay factor results in better performance but fewer overall predictions.

Decay Factor (β): To determine the choice of decay factor β, we set β to values ranging from 0.0 to 1.0 and observe the performance of the approach. In addition to β, we set the number of hops to 2 and $\Delta = 1$ week. The result of this experiment is shown in Figure 4.8. Here we also show the rate of $pc = \frac{pred}{count}$ which measures the proportion of times that the semantic graph-based approach could predict at least one hashtag for the tweets in the test set. We see that a smaller decay factor results in a better performance but fewer overall predictions. Hence, to balance these two factors, we set $\beta = 0.8$.

Δ T	Precision	Recall	F-measure
hourly	0.041	0.042	0.041
hourly4	0.038	0.040	0.039
daily	0.109	0.107	0.107
weekly	**0.174**	**0.261**	**0.203**
monthly	0.132	0.201	0.152

Table 4.3: Comparing AR predictions with different ΔT. The weekly sliding window builds a better prediction model.

Length of Sliding Window (Δ): We additionally repeated the experiments by varying the length Δ to different values. We set the number of hops to 2 and parameter $\beta = 0.8$. The result of this experiment is shown in Table 4.3. We see 1 week of sliding window gives the best performance. In comparison, we see that the hourly, 4 hours and daily windows are sparse resulting in poor performance, while a month of data tends to recommend outdated hashtags which also results in poor performance.

Approach to Select Pivot Terms: As described in Section 4.2.2.2, an important problem in the graph-based prediction framework is to select correct pivot terms. We now evaluate the performance of our approach using the two methods – entropy and

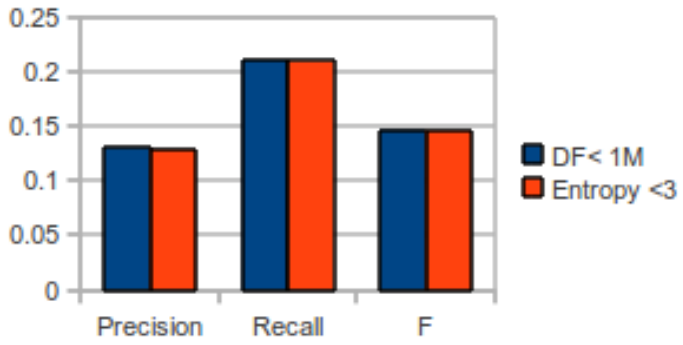

Figure 4.9: DF and Entropy pivot selection perform nearly equally well.

document frequency. The results are shown in Figure 4.9. Interestingly, we observe little difference between the performance of these two approaches. Since document frequency is simpler to maintain for all terms, we select it for the remainder of the experiments.

4.3.3.2 Comparison of Annotation Prediction Methods

We next evaluate the effectiveness of the several alternative methods for predicting hashtags. We then present the results of comparing our graph-based approach in detail against the best of these alternative methods.

Comparison of Alternate Methods: The comparison between annotating approaches in [85], [94], and [29], described earlier in this section, is shown in Table 4.4. For association rules, we report results for $conf = 0.1$ and $sup = 30$; we additionally varied the support threshold between 10 and 100 but found little change in results. We see that the association rule approach results in the best precision, recall, and F-measure (it also is relatively more efficient than the alternate approaches). Intuitively, the association rule approach is effective at weeding out large numbers of weak term-hashtag pairs (via the confidence and support thresholds), resulting in

Method	Precision	Recall	F-measure
promo-vote	0.008	0.025	0.011
promo-sum	0.004	0.014	0.006
bayes	0.023	0.039	0.027
assoc-rule	**0.167**	**0.218**	**0.189**

Table 4.4: Comparing alternative approaches over 1000 test tweets.

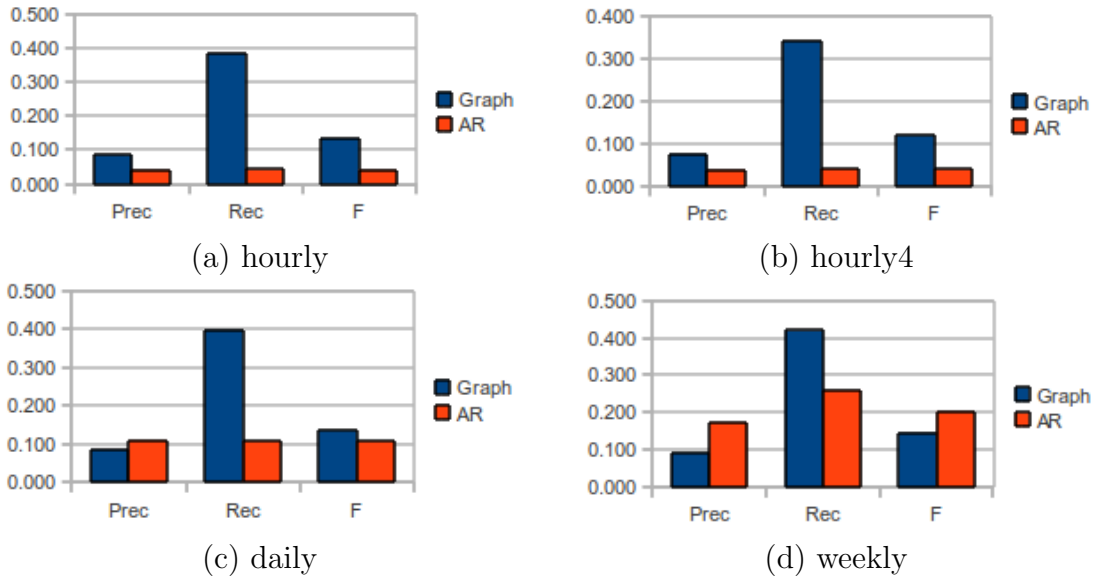

Figure 4.10: Comparing the graph-based and Association Rule based models for different sliding windows. The graph-based approach achieves high recall in all cases and better precision for the shorter sliding windows. The AR approach works well over the longest time horizon, when the training set is the largest.

the best relative performance.

Graph-based vs Association Rule: Since association rule mining approach performs the best among the alternate approaches, we now compare it with our graph-based proposed method. Figure 4.10 compares the association rule based model and the graph-based approach for windows of different lengths. We observe that the association rule approach gives good performance when the length of the sliding windows is large (since it has access to a larger training set to identify term-hashtag relation-

ships). However the graph-based model has a higher recall in all cases and better precision for the shorter sliding windows. These results suggest that the graph-based approach can identify implicit relationships among terms and hashtags by linking terms and hashtags that may have never occurred together.

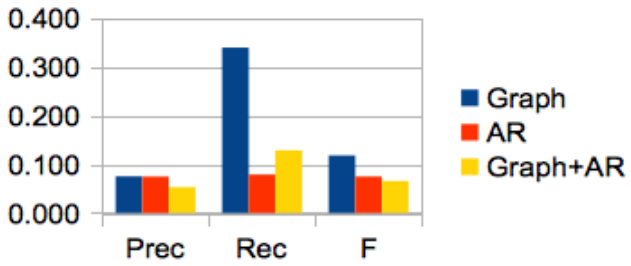

Figure 4.11: Combining AR with the semantic graph improves recall but not precision.

Combining AR and Graph-Based Approaches: A possible extension of the association rule based model is to combine it with the graph-based annotation prediction method. In this way we could take advantage of the properties of the graph-based model for revealing implicit relationships. Hence, we augmented the term-hashtag association rules discovered by association rule mining by additionally scoring related hashtags using the graph-based approach. In this way, additional hashtags may be identified, offering the possibility of increased recall. We evaluated the performance of this extended version and report the results in Figure 4.11. While we do observe that the recall of the combined approach is higher than the baseline association rule approach, it is still less than the pure graph-based approach. And disappointingly, the precision of the combined approach is worse than either alternative, suggesting the need for careful future study of the combination of these two approaches.

4.3.3.3 Analysis of Graph-Based Approach

Finally, we turn our attention to analyzing several properties of the graph-based prediction approach and describe a technique to extend its performance using tweet and hashtag categorization.

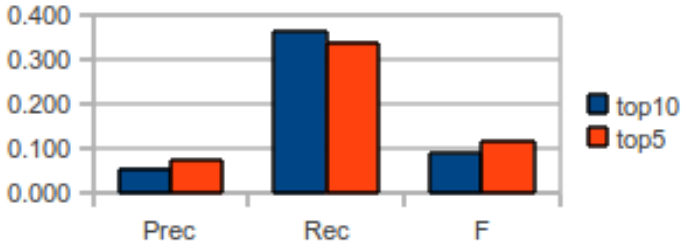

Figure 4.12: Increasing the number of selected hashtags (topK) lowers precision and increases recall.

Impact of Number of Hashtags: Based on the scores for hashtags generated by our system we select the first *top-K* hashtags. We observe that when *top-K* is small, we have higher precision and when it is larger we have higher recall. We consider *top-K* = 5 for the experiments since it gives us a good balance of precision and recall.

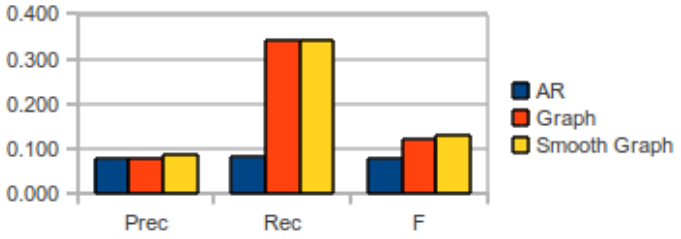

Figure 4.13: Smoothing increases precision by incorporating longer-term term-hashtag relationships.

Impact of Smoothing: In Section 4.2, we described a smoothing model considering a mixture of both the most recent window and the global history in terms of hashtag-

term linkages in the semantic graph. Performance of this smoothed model with others is shown in Figure 4.13. We observe a small increase in precision, but almost no improvement in recall. Additionally, since we are dealing with more data in the smoothed approach, the time taken to build model is greater, which may be infeasible for real-time annotation as status updates are inserted into the system.

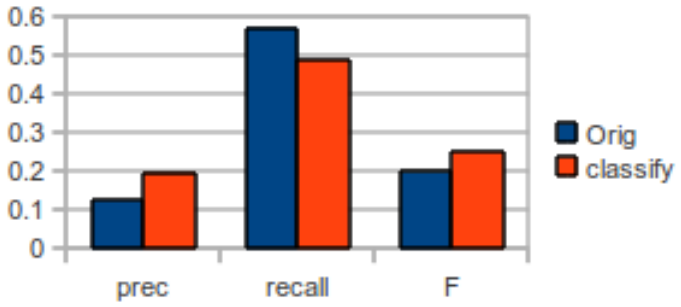

Figure 4.14: Classification of tweets increases the performance of the baseline approach.

Extending the Approach with Categorical Information: So far, we have studied semantic annotation of status updates using only the content of the updates themselves, without access to additional meta-information about the updates. It may be reasonable to expect that incorporating the category of the update into the prediction framework could increase its performance. Hence, we explore the possibility of improving the predictor by filtering out all suggested hashtags that belong to categories other than the category of the status update itself. Towards this goal, we assume there exists a tweet classifier similar to what is proposed in [80, 88] that can categorize both tweets and hashtags. Here we use the top-500 frequent hashtags that are already labeled by [80] into 8 categories: *Celebrities, Game, Political, Idioms, Music, Movies, Sports, Technology*. Then we consider only the tweets that contain at least one of these labeled hashtags (resulting in 12 million tweets in the dataset).

Figure 4.14 compares this categorical extension with the baseline graph-based model. As expected, we see an increase in precision for the categorical extension, but a decrease in recall. This suggests the potential for incorporating more refined categorical (and perhaps sentiment-based) information into the hashtag prediction framework.

4.4 Conclusion

In this chapter, we proposed a graph-based prediction framework for increasing the coverage of semantic annotations in real-time web status updates. We saw how the path aggregation technique for scoring the closeness of terms and hashtags in the graph, pivot term selection, and the dynamic sliding window led to encouraging results in comparison with alternative methods. As systems like Twitter and Facebook continue to grow, the proposed approach could be used to extend the small fraction of self-curated messages to organize the vast majority of messages that have not been annotated. In this way, the feedback between small-scale curation and automated methods may provide an evolving framework for ongoing organization of real-time web content.

5. SUMMARIZING SHORT REVIEWS: A CLUSTER-BASED APPROACH*

5.1 Introduction

In total, user-contributed short text can convey the aggregate opinion of communities about political discussions, web media, products, and other entities of interest. This short text can then be used by others to improve their understanding of current issues such as, what topics are important? what arguments are being made?, for a product, what aspects of this product do customers enjoy?, and for expanding their horizon of viewpoints. Ultimately, the aggregated opinion of communities expressed through short text could lead to changed political decisions, purchasing decisions, and impact a person's view of the world. However, an interested party may need to scroll through many short text items to synthesize the main points; for example, one might need to scroll through dozens of pages of comments with repeated statements, duplicate content, and low-quality short text to find such information and not all the users have the patience and time to go through all comments and reviews. Hence, while higher level of user engagement with online media would provide more information for the users, it is difficult to identify the critical information that a user needs to grasp from the entirety of available short text.

Toward overcoming this challenge, we study in this chapter the short text data summarization problem: for a set of n user-contributed short text associated with an online resource, select the best top-k representative short text for summarization. Unlike traditional multi-document summarization which has typically focused on high-quality documents in relatively small collections, short text summarization faces

*Reprinted with permission from "Summarizing User-Contributed Comments" by Elham Khabiri, James Caverlee and Chiao-Fang Hsu, 2011. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, 534-537, Copyright 2011 by The AAAI Press.

many unique challenges: e.g., high-variability in short text quality, wide ranging short text lengths (from one or two words to many paragraphs), multiple competing opinions, implicit references to earlier short texts, and so forth. The final goal of this chapter is to help users digest the vast diversity of opinions in an easy manner.

In this chapter we first give a background of different types of summarization methods on identification of representative sentences and grouping similar short text together. Then we introduce the proposed algorithms and intuition behind them. We then apply the proposed algorithms on three types of datasets with diverse natures. Different evaluation metrics will be introduced and the proposed methods are applied to each of them separately.

5.2 Background

In traditional definition of summarization, a summary is explained as a text that is produced out of one or more input texts, containing some of the same information of the original text and is no longer than half of the input text [32]; however, in the social media we are facing two types of summarization for the user contributed input texts. First, *structured summary*, which is the number of votes for a web object or the number of votes for each specific aspect of an object. As an example, for a product in Amazon.com, with hundreds of comments contributed by many users, we can see a diagram, that shows how many of the users have given 5-stars and how many voted for 4-stars and so on. Although such structured summaries are useful and quick to comprehend, they are not sufficient to reflect the huge amount of available content from hundreds and thousands of short text contributions. Therefore, there is a need to provide the web users with *textual summaries* so that they can learn the reason behind the number of received votes. *Opinion summary* and *social summary* are two types of textual summaries. In opinion summary, the focus is on extracting the

aspects and their relevant sentiments. Social summary on the other hand, is very close to the traditional definition of the summary mentioned above. It does not intend to apply any hard lines for containing different aspects. Instead, it include the sentences that have received high social attention from the web users. Although it may not include all the aspects that are available for an object, it extracts the sentences that are discussed by many of the web users and hence, are deserved to be included in the summary. In this chapter we propose, discuss and evaluate different algorithms on creating such social summaries.

5.3 Overall Approach: Identifying Representative Sentences

Our overall goal is to select the most representative and informative short texts from a large collection of user-contributed data. At the same time the selected sentences should cover different viewpoints about the resource that can highlight various aspects of it. We define V as the set of all resources that we have in our dataset $V = \{v_1, v_2, ..., v_n\}$. Each resource v_i is associated with a set of sentences $C_i = \{c_1, c_2, ..., c_m\}$, where each c_j is a single sentence that we consider it as a bag of words. Here m is the total number of sentences. Our goal is to extract a subset of C_i, $S_{C_i} \subset C_i$, that are the most representative sentences: $S_{C_i} = \{s_1, s_2, ..., s_n\}$. We have a ranking of all of the sentences for each resource and n is a tunable parameter. For example if $n = 5$ we select the top-5 sentences from the ranked list of available short text data. Since our goal is to summarize a large set of sentences for quick understanding, we will typically require $n \leq 5$, though larger values may be appropriate in some situations. We investigate different methods to obtain the most representative comments submitted by the web community. Our overall approach is:

1. Identify groups of thematically-related sentences.

2. Rank groups according to a measure of significance.

3. Rank sentences within each group according to a measure of importance.

4. Select sentences from each group.

In this approach first we apply a high level clustering of all the sentences of each resource and then select the most informative ones out of each of such clusters. Figure 5.1 shows that the first step is to construct clusters similar short text out of all the input data. Then for each cluster we rank it based on the importance score. Finally we sort clusters based on the size. Bigger size means there are more sentences talking about the larger topic. Then we pick the first most important sentence from the first sorted cluster and then the first most important sentence from the second sorted cluster. After picking the first important sentence out of each cluster, we continue to pick the second important sentence from each cluster. We repeat this process until we reach to the limit of the summary size which is a parameter defined by the user.

5.4 Identify Groups of Related Sentences

From a diverse available clustering algorithm we examine the two most well-known algorithms in the area of unsupervised text clustering to identify thematically similar sentences: topic-based clustering and k-means clustering.

5.4.1 Topic Model-based Clustering

A topic model is a generative probabilistic model which uses a distribution of vocabularies to identify the underlying topics that documents are generated from. It considers the co-occurrences of terms and their frequencies to cluster similar documents into thematically similar groups. For example, if terms $\{river, bank\}$ co-occur regularly, while terms $\{river, account\}$ never co-occur together, then we can assume that there is one topic including terms *river* and *bank*, and there is a different topic including *account*. Well-known topic modeling approaches are Probabilistic Latent

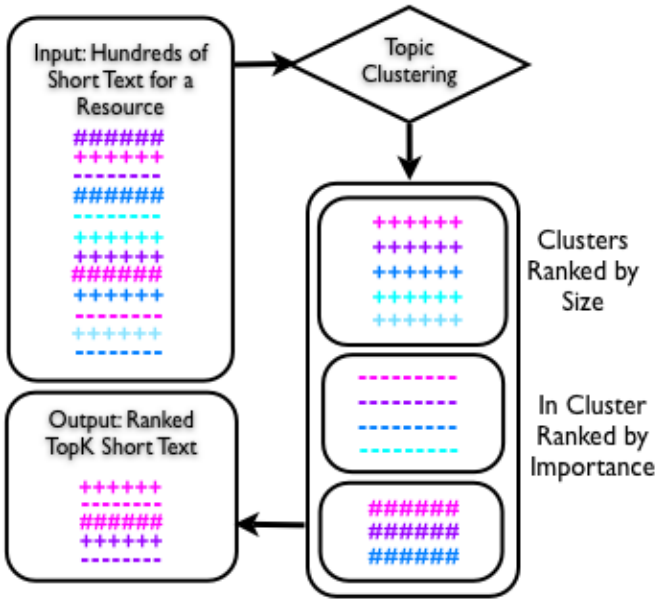

Figure 5.1: The overall summarization architecture: The input consists of combination of sentences with different topics and quality. Higher quality is shown as pink and lower quality as blue. Different styles represent different topics. The output consists of a variety of topic of high quality.

Semantic Analysis (PLSA) [31] and Latent Dirichlet Analysis (LDA) [7].

The goal of topic modeling is to identify a set of topics or themes from a large collections of documents. Based on the probability of topics, such models try to identify documents that are relevant to each of the themes. For example, in a YouTube video showing President Obama after an election speech, possible themes of the comments include his talk content, comparisons with his opponent Mitt Romney, and discussions of the way he and his family are dressed. Another example would be reviews about a laptop product. Possible themes are battery life, portability, cost and appearance that are viewed as different aspects of a product. LDA and PLSA are similar in the sense that they both view each document as a mixture of various topics. However, in LDA the topic distribution is assumed to have a Dirichlet prior rather than a uniform distribution. Therefore we have a more reasonable mixture of

topics in a document with LDA [24]. The other problem with PLSA is that there is not a direct way to apply the learned model to the new document, while in LDA you can use inference to cluster new samples to existing clusters. Moreover, when the number of documents increases, PLSA suffers from the overfitting problem [92]. With all the above reasons we adopt the LDA model. Particularly we use LDA to extract T topics out of all the sentences associated with a single resource. The number of the clusters is assigned manually.

For each document in a corpus of M documents, LDA assumes the following generative process.

1. Choose distribution of latent topics θ from $Dir(\alpha)$
2. Choose distribution of words ϕ from $Dir(\beta)$
3. To generate each document, for each word:
    - Choose a topic z from $Multinomial(\theta)$
    - Choose a word w from $Multinomial(\phi)$

Figure 5.2 shows the dependencies among the observable and latent variables used in LDA in a standard graphical model plate notation where boxes are called "plates" and are representative of replicates. The outer plate represents documents, and the inner plate represents the repeated choice of topics and words within a document. M denotes the number of documents, and N the number of words in a document. There is only two parameters α and β as the Dirichlet prior per-document topic and per-topic word distributions. Words are the only observable variables and the rest are all latent variables.

In LDA we can incorporate the existing knowledge as prior probabilities. For example, in the products reviews, the prior knowledge of the topics or the sentiment

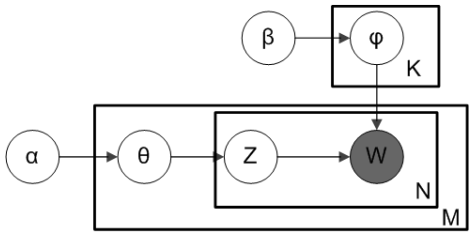

Figure 5.2: Plate notation for LDA

orientation about the product results in more reasonable clusters [99]. However, this needs human experts to define rules for separation and merging of the clusters with specific words.

Back to the clustering problem, we transform the LDA model with assignment of distribution of topics to each document from soft clustering to a hard clustering using Maximum Likelihood. In the LDA original form, we have a set of documents $D = \{d_1, d_2, ..., d_n\}$ and a number of topics $T = \{t_1, ..., t_m\}$ that are considered as clusters for the documents. Any document d_i can be explained by any topic sequence with some likelihood. For example, $Pr(d_1 \in t_1) = 0.70$ and $Pr(d_1 \in t_2) = 0.20$ and so on. In the topic model-based clustering we have considered each short text unit as one document and then we found the maximum a posteriori topic for each document.

$$r = argmax_r Pr(t_r|c) = argmax_r Pr(c|t_r)Pr(t_r) \tag{5.1}$$

Therefore r is the topic number that has the maximum likelihood for each document. In the above example we say that d_1 belongs to t_1. The granularity of each document ranges from the whole short text for a resource, to the single short text such as comments or reviews or even smaller units such as sentences or phrases used in the short text data. Table 5.1 shows a sample of the topics extracted from all the comments of all the videos in a YouTube dataset. As it is shown, each cluster is

concentrated on one specific topic. The general meaning of each cluster is written in parentheses.

topic 1 (music)	topic 2 (killing)	topic 3 (election)	topic 4 (woman)	topic 5 (game)
song	kill	peopl	woman	team
band	shoot	republican	she	game
music	school	govern	women	sox
singer	eric	support	girl	player
jimmi	kid	bush	beauti	fan
rock	psychopath	huck	bitch	walton
album	bomb	candid	dress	nba
sound	killer	elect	free	mj
roll	peopl	liber	sarah	season
webster	shot	main	leg	greatest

Table 5.1: Topics extracted from YouTube comments. All the comments for one video is considered as one document.

We next illustrate how the topic clustering algorithm groups similar comments of a resource together. In the video "Young atheist on Wife Swap", two wives with two different viewpoints (liberal and conservative) participate in the Wife Swap program[†]. The LDA-based algorithm groups comments from two different viewpoints (*Religion* and *Patriotism*) into two separate clusters. In Table 5.2 comments for each of these clusters are quoted:

5.4.2 K-Means Clustering

As an alternative, we consider the k-means clustering algorithm, in which each short text unit (sentence or paragraph) is a vector model with size M, where M is the size of the dictionary $X = [x_1, x_2, ..., x_M]$. Each of the x_i is the *tf-idf* score of the term used in the short text. The k-Means algorithm assigns each short text to the cluster whose center is nearest. The center is the average of all the vector models in that cluster. The algorithm includes:

[†]http://www.YouTube.com/watch?v=1CIhn3wPFnE

Cluster 1 (Religion Aspect)	Cluster 2 (Patriotic Aspect)
"I hate how these kinds of atheists give the rest of us atheists and agnostics a bad name. I'm an agnostic but I don't believe in extremism."	"if dan doesnt think the US is the greatest country in the world thats fine....because the US gives him the right to get the F* out."
"Atheists accept anything with scientific reasoning, that doesn't make them narrow minded"	"I love his reaction to "America is the greatest country in the world" comment :P"
"I don't know how he can have so much hatred I see no hatred. I see a guy explaining why he doesn't believe in Christianity. I guess that means non-Christians are all just full of hate." "Saying there is not a God is Idiotic? What, then, is the word youd use to describe someone who talks to an invisible man in the sky?"	"You are a f*** retard. Tell me how is Usa no 1 country rate whit such a high crime rate, teen pregnancy, mortality, in health care you are on 55th, most free country is Netherlands, on education you are 72th so please tell me how can you be no 1 if you are 55th in this category and 72th on other. You need to stop braging about early generations it is 2009 and not 1950's by saying that you lose your moral high grounds."

Table 5.2: Extracted topics from YouTube video comments. Each comment is used as a document.

1. Choose k as the number of clusters.

2. Randomly generate k random cluster centers or centroids.

3. Assign each short text to the nearest cluster center.

4. Recompute the new cluster centers.

5. Repeat the two previous steps until the assignment is not changed and convergence is met.

In the experiments section we study the two variations of K-Means clustering, one is based on all the words and one is only based on the nouns appeared in the short text. We compare these variations with the ones in the topic generative model.

5.5 Identifying Significant Short Text inside Clusters

After producing our clusters, the next step is to select the most informative short text in each of them. Users want to focus immediately on a handful of key sentences that communicate the key ideas from across all short text. We need some way of selecting one or a handful of short text per cluster. That is, given a cluster, select a sentence (or a few) that best expresses the cluster. We consider two approaches: a term-importance based approach and a PageRank based approach.

5.5.1 Term Importance

The first approach to ranking short text within a cluster is by awarding more points to sentences containing "important" terms. The intuition is that sentences containing more significant terms are themselves more significant. We consider two approaches to term importance: a vector space (geometric) measure and an information theoretic measure of term importance. In selecting sentences by vector space-based importance, $tf_{i,j}$ is defined as the number of times a $term_i$ appears in the sentences of a particular $resource_j$ normalized by the total number of terms in the sentences of that resource, and idf_i is the logarithm of total number of resources $|D|$ divided by the number of resources that $term_i$ appeared in.

$$\mathrm{tf}_{\mathrm{t_i,v_j}} = \frac{n_{t_i,v_j}}{\sum_{t_k \in v_j} n_{t_k,v_j}} \tag{5.2}$$

$$idf_{t_i} = \log \frac{|D|}{|\{v_d : t_i \in v_d\}|} \tag{5.3}$$

The importance of each sentence c_k is the average of the importance of terms used in that short text using tf-idf metric.

$$tf\text{-}idf_{c_k,v_j} = \frac{\sum_{t_i \in c_k} tf_{t_i,v_j} \times idf_{t_i}}{|c_k|} \tag{5.4}$$

In selecting sentences by information theoretic importance, we measure how much information the presence or absence of a term contributes to the term appearing in the appropriate cluster. For each term of sentence i in cluster k, $c_{i,k}$, we calculate the Mutual Information (MI) of that term. Formally we define MI of term t and cluster k as:

$$MI(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \frac{p(x,y)}{p_1(x)\, p_2(y)} \tag{5.5}$$

X is a random variable that shows if a sentence contains term t or not, Y is a random variable that means if the sentence is in cluster k or not. We want to know how much being in cluster k depends on having term t in the short text.

$$\begin{aligned} MI(X;Y) = & \frac{N_{11}}{N} \log_2 \frac{N_{11}N}{N_{1.}N_{.1}} + \frac{N_{10}}{N} \log_2 \frac{N_{10}N}{N_{1.}N_{.0}} + \\ & \frac{N_{01}}{N} \log_2 \frac{N_{01}N}{N_{0.}N_{.1}} + \frac{N_{00}}{N} \log_2 \frac{N_{00}N}{N_{0.}N_{.0}} \end{aligned}$$

The first subscript indicates if a sentence contains the term or not and the second subscript shows if we are considering the sentences in the current cluster or other clusters. N is the total number of sentences. Suppose that we want to find the MI of the term t of a resource which is grouped in cluster k. N_{10} shows the number of the sentences that contain term t and are not in cluster k. Finally we rank the

sentences by the average MI of all the terms in each sentence.

$$MI(c_{i,k}) = \frac{\sum_{t \in c_{i,k}} MI(t;k)}{|w|} \tag{5.6}$$

Note that we have used sentence as the short text unit in the above definitions. This has been used in the experiments about product reviews. For the short comments, since each comment is composed of one or very few number of sentence, we consider the whole comment as a short text unit.

5.5.2 PageRank based Ranking

In recent years, the graph-based ranking methods, including TextRank [66] and LexRank [19] have been proposed for document summarization. Similar to Google's PageRank algorithm [73] or Kleinberg's HITS algorithm [43], these methods first build a graph based on the similarity relationships among the sentences in a document and then the importance of a sentence is determined by taking into account the global information on the graph recursively.

Let the set of short text sentences be $S = \{s_1, s_2, ..., s_n\}$, where n is the total number of sentences for a resource. Here, s_i is represented by a bag of words, i.e., $s_i = \{t_1, t_2, ..., t_m\}$, where m is the number of distinct non-stop words. We define a graph $G = (V, E)$ for all the sentences related to a resource, in which the nodes from V are the sentences that are connected through edges $e_{ij} \in E$. There is a link between two nodes if the similarity of the sentences are more than a threshold.

One hypothesis that we want to examine is if there is a relation between the sentences of a resource. What does it mean when the later sentences are talking about the similar same issues that the earlier sentences have talked about. Does it mean as an endorsement of the earlier sentence when the later one is using similar terms or not? Performing a random walk, we can find sentences with more "vote of

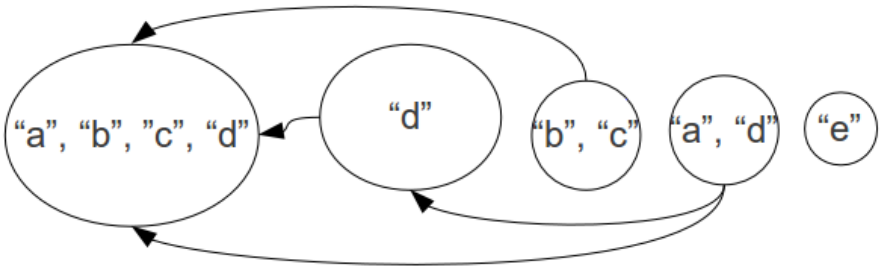

Figure 5.3: Nodes and links of sentences in one resource. The left sentences has appeared first.

support” from the later sentences. Using such PageRank algorithm, we aim to select the sentences that receive highest number of in-links among hundreds of other ones. See Figure 5.3. The random walk continues iteratively until the scores of the nodes are converged.

To calculate the PageRank score of a sentence $PR(s_i)$ we add the score of all the neighbors pointing to it divided by the number of output links of each of these neighbors. We used 0.85 as our damping factor α.

$$PR(s_i) = \alpha \times \sum_{s_j \in N(s_i)} \frac{PR(s_j)}{outlink(s_j)} + (1 - \alpha) \tag{5.7}$$

The N_{s_i} is the set of neighbors for the s_i. The $outlink(s_j)$ is the number of outlinks for the neighbor s_j. In the process of constructing the graph, we do not consider the edge weights, i.e., the nodes are simply connected if the number of common terms are greater than a threshold. One variation would be to consider a weighted graph in which the edge weights are measured by any similarity metrics such as, raw number of common terms, normalized number of common terms, Jaccard coefficient, or cosine similarity. For example, if $S_1 = \{a, b, c\}$ and $S_2 = \{b, c, d\}$ the raw common count will be $|S_1 \cap S_2|$=2 and the Jaccard coefficient will be $\frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} = \frac{2}{4}$. In the experiment,

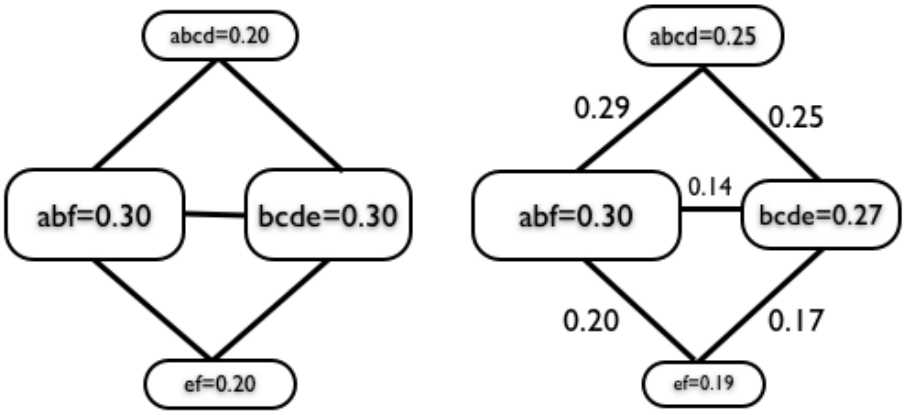

Figure 5.4: The left graph does not consider the weight between the nodes. Therefore, both "ef" and "abcd" receive same scores regardless of the extent of similarities to their neighbors. This is not the case for weighted PageRank.

we found that even the raw count gives a reasonable similarity measure since we only consider the meaningful terms that are already preprocessed by removing stopwords. The Weighted PageRank is measured as:

$$WPR(s_i) = \alpha \times \sum_{s_j \in N_{s_i}} WPR(s_j) \frac{W_{out}^{(ji)}}{\sum_{s_k \in N_{s_j}} W_{out}^{(jk)}} + (1 - \alpha) \tag{5.8}$$

The N_{s_i} is the set of neighbors for the s_i. $W_{out}^{(ji)}$ is the weight of outlink edges from j to i. Figure 5.4 shows how the weighted edges affect the score of each node. The edges are weighted based on the Jaccard Coefficient. Larger edge weight will result in a larger inlink effect.

Another variation is to consider the importance of the common terms between the nodes. So that the more informative common terms will result in a higher score. The *tf-idf* is a good candidate for this purpose. T_{common}^{ij} is the summation of the

tf-idf scores of all terms that are in common between two nodes i and j.

$$PRT(s_i) = \alpha \times \sum_{s_j \in N(s_i)} \frac{PRT(s_j)}{outlink(s_j)} * T^{ij}_{common} + (1 - \alpha) \tag{5.9}$$

$$WPRT(s_i) = \alpha \times \sum_{s_j \in N_{s_i}} WPRT(s_j) \frac{W^{(ji)}_{out}}{\sum_{s_k \in N_{s_j}} W^{(jk)}_{out}} * T^{ij}_{common} + (1 - \alpha) \tag{5.10}$$

Both the weighted and non-weighted versions of PageRank are shown considering the *tf-idf* of the common terms.

5.6 Experiments

In this section, we present an experimental study of short text summarization over a collection of short text data relevant to different kinds of web resources such as a video or an online product. First, we evaluate cluster based approach with all of its parameters and variations, then we evaluate the PageRank based ranking. Finally we show that the combination of these two gives us promising results. This happens for different perspectives that each method suggests; cluster based ranking takes into account the variety of results besides more focus on the short text that are more distinctive and informative in each cluster, and PageRank based ranking focus more on the term usage of the short text that have attracted more attention for many users.

5.6.1 Dataset

For the experiments we use the comments of YouTube videos and the reviews of products in two well-known review websites, Amazon and CNET.

For YouTube dataset we crawled title, related user generated tags, video category and comments of 17,600 videos from the YouTube website using Tubekit [84]; Figure 5.5 shows an example. Established in 2005, YouTube now constitutes more than 10%

of all traffic on the Internet [13] and accounts for approximately 60% of the videos watched on the Internet. At its current rate, about 65,000 new videos are uploaded per day [79] amounting to 24 hours worth of video per minute [1]. To make sense of this mass amount of content, one of YouTube's community collaboration tools is commenting. The analysis of the comments constitutes a potentially valuable data source to obtain implicit knowledge about videos. In the YouTube website, each video consists of the following fields: 1) Title, assigned by the poster of video, contains a short topic about it; 2) Description, assigned by the poster of video, contains short explanation about it; 3) Keywords, assigned by the poster of video, contains tags that are related it; 4) Comments, assigned by the web community, contain their knowledge or idea about the video. Figure 5.5 shows an example. To illustrate the potential of comments, we find that over a collection of 17,600 videos that there are 4.7, 53.3, 25.3, words in title, keyword, description respectively and 434.2 words in all the comments for each video, indicating that comments provide a potentially rich source of contextual information about a web object. To sample videos, we issued queries drawn from two different policies: (1) Random word selection from an English dictionary resulting in 240 queries; and (2) The top most popular queries based on Google trends from September to November 2009, resulting in 3,596 queries.

Title: Toy Story 3
Release Date: 18 June 2010
Genre: Animation
Cast: Tom Hanks, Tim Allen, John Ratzenberger, Joan Cusack, Michael Keaton
Director: Lee Unkrich
Writers: Michael Arndt
Studio: Walt Disney Pictures

Plot:
Woody, Buzz, and the rest of their toy-box friends are dumped in a day-care center after their owner, Andy, departs for college.
Subscribe Now: http://www.youtube.com/subscription_c...

Category:
Entertainment

Tags:
Broadbandtv viso film movie clips story video media show cinema theatre

All Comments (16,524)

abbeycoffee
5 minutes ago

This has been flagged as spam hide
Nice! i can watch this for free in HD
watchonlinemoviesfree(dot)org
.
watchonlinemoviesfree(dot)org

yusgayus
13 minutes ago

This time however, the toys' owner Andy is now 17, about to go off to college and no longer as close to his old friends as he used to be. Indeed, many of the toy characters from the previous films have been sold or

Figure 5.5: Snapshot of a video from YouTube

The other kind of dataset is the comments related to the products in product review websites. From the 20,423 products crawled from Amazon [8], we selected those products with at least 6 reviews on a product. This resulted in 21 different

categories with total of 3,679 products and 29,884 total reviews. The number of reviews ranges between 6 to 12 and the number of sentences ranges between 11 to 305 sentences per product. On the other hand, Amazon has recently added a new feature called *highlights* which is a 3 lines of summarization for available reviews generated by Amazon. We crawled these *highlights* to compare our summarizer's outputs with them in the experiments. Figure 5.6 shows a snapshot of the Amazon website.

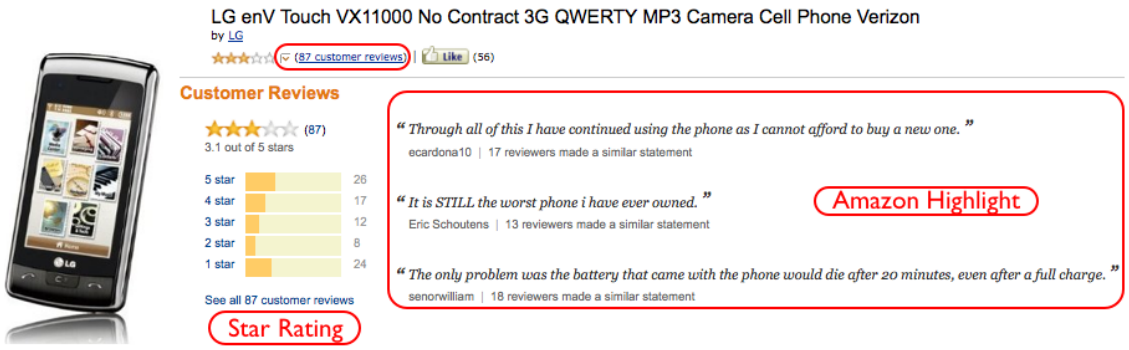

Figure 5.6: Amazon offers different quality measures by the users using star rating and the highlights.

The third dataset is CNET reviews [22] for 330 different products with at least 5 associated reviews. In CNET, the user can write a full text review of a product and summarize their pros and cons about that product in the separate section. In one example, a user has talked about his positive experience about a product in the review section, while in the pros section he has mentioned *"everything"* and in cons section he has mentioned *"nothing"*. Although these pros and cons gives an overview of the content submitted by a user, they are not enough to capture the gist of it. Therefore, showing a scalable summary would be beneficial for the web users. Figure 5.7 shows a snapshot of the CNET website.

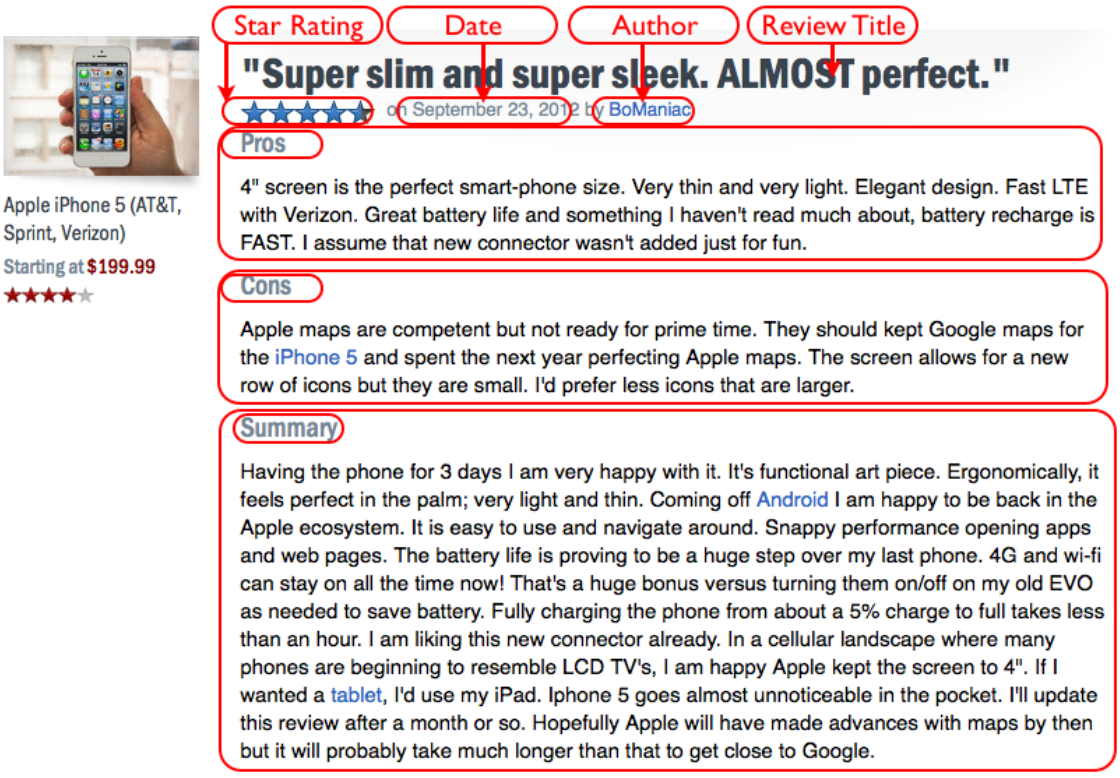

Figure 5.7: CNET is a product review website with reviews and pros and cons.

5.6.2 Evaluation

Generally it is a very difficult task to compare which of the output summaries is the best among many of different choices of summarization. Even if it is done by human judges, we do not always have 100% agreement among them. The judges' idea about the usefulness of a summary might be a factor of their personal experiences and preferences. On the other hand, we have automatic evaluation of summaries which is based on some heuristics of the goodness of a summary. Although automatic summarization is not as good as human evaluation, its results could be similar to it. As what [57] showed, the information-theoretic based measures are highly correlated with human evaluations.

5.6.2.1 Human Evaluation

In the ideal world, we would like to have human judges who spend time to tell us which of the output summaries make more sense. Although this sort of evaluation is an expensive choice, but it is more reliable compared to automatic evaluation. In our experiments, we had to limit the number of user based evaluations. Therefore, we only used YouTube dataset for user based evaluations. To evaluate different algorithms, we conducted user studies on 5 subjects and 30 videos. The selected videos received between 500-1000 comments each. To make evaluation possible we selected the first 50 comments out of each videos and showed them to our human subjects. We asked them to mark the comments which they found interesting and informative. By aggregating the number of times each comment was selected, for each comment we have a score between 0-5. Score 5 means all of the 5 human subjects found the comment interesting and score 0 means none of the human subjects found it interesting. We used the well-known method normalized discounted cumulative gain (NDCG) that checks if highly relevant content has appeared earlier in the ranking results.

$$DCG(P) = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i)} \tag{5.11}$$

$$NDCG_p = \frac{DCG_p}{IDCG_p} \tag{5.12}$$

$IDCG$ is the ideal ranking in which the most relevant result has appeared in the first position of ranking. The second most relevant is appeared in the second position and so on. $NDCG$ shows how much we are close to the ideal ranking.

5.6.2.2 Automatic Evaluation

Inspired by [57], we used the KL-divergence to measure how well a summary could capture the gist of whole input text. Figure 5.8 shows the distribution of different terms in all the short texts relevant to a resource and two summaries. The green summary has the same distribution of terms as the input text. The red summary on the other hand, shows a high divergence from the original input text and is not considered as a good summary.

$$KL(I(x)||S(x)) = \sum_{x \in Dic} I(x) \log \frac{I(x)}{S(x)} \tag{5.13}$$

I is the distribution of terms in input text and S is the distribution of terms in summary. If KL-divergence is near zero, then it means that the two distributions are highly correlated. Higher value of KL-divergence shows more distance between the distributions of terms.

We have 2 more measures to evaluate summaries automatically without any human intervention: Compression Rate (CR) and Retention Rate (RR). A good summary should be short enough and yet it should retain the information of the input text as much as possible [32].

$$CR = \frac{length(Summary)}{length(Input)} \tag{5.14}$$

$$RR = \frac{info(Summary)}{info(Input)} \tag{5.15}$$

There is a trade off between Compression Rate and Retention Rate. A desirable summary will have a low CR and high RR. For our experiments, we evaluate the RR only, since we the length of output is fixed to a certain length. Therefore, for a

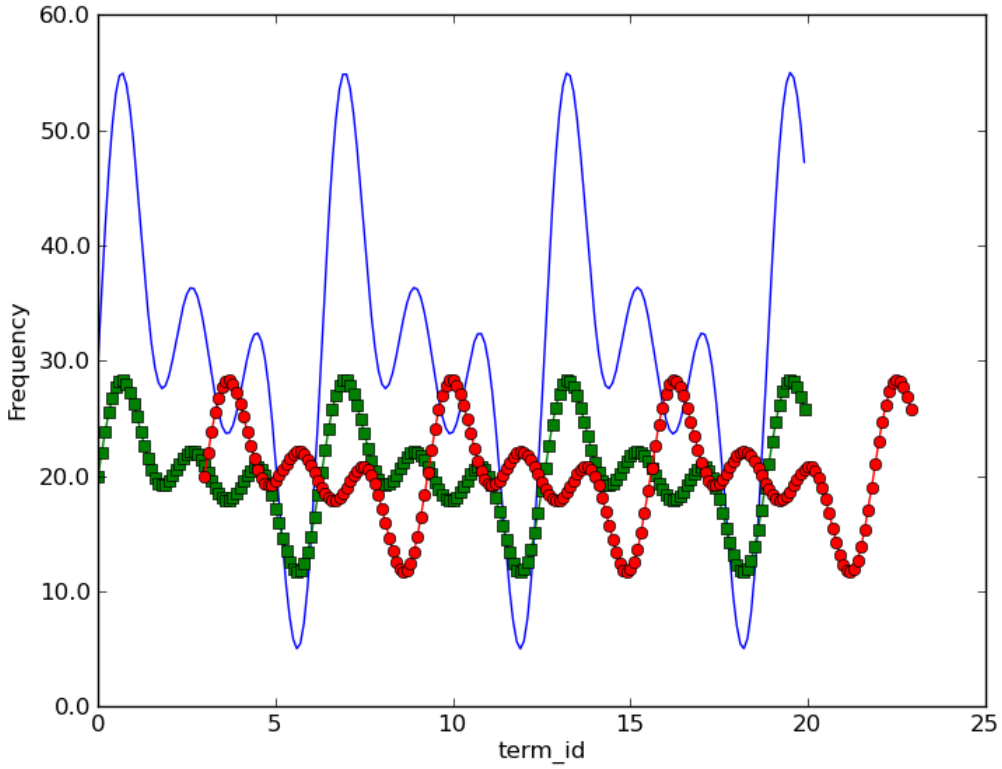

Figure 5.8: Example of a good and a bad summary for an input text. A good summary will have similar distribution of terms as the input text.

fixed length, we would like to measure the amount of non-stopword terms that are found in the summary.

5.6.2.3 Alternative Methods

There are different alternative methods that we compare them all together. We are interested in knowing which of them are more suitable for the task of social summarization and if a method works better for one type of dataset rather than another.

- PR: The PageRank score of each sentence in all the comments pertaining a resource.
- $PR + LDA$: The combination of LDA clustering method and PageRank score.
- $MEAD$ and LR: Mead and LexRank [76, 19] scores that are graph-based ranking methods. These two methods are used as baselines in the experiments related to human based evaluation.
- $MEAD + LDA$: The combination of MEAD and LDA clustering.
- $LR + LDA$: The combination of LexRank and LDA clustering.
- $TFIDF$: The average of *tf-idf* score for all the terms used in a sentence. We would like to value the terms that are not too general and at the same time have a higher score for the terms that are repeated in the sentences for the target product.
- $RANDOM$: Extract few sentences out of all the input sentences randomly. This approach is used as a baseline in the experiments.

As a pre-processing step, we apply a part of speech tagging on the input sentences to make sure each one contains at least one noun and one adjective. Reason for it is

that there are a lot of non-informative sentences that are not giving any sentiment (adjective) for specific aspect (noun) of a resource (either a video or a product in our case). For example, the sentence, "I bought this speaker for my son's birthday", is not relevant to any of the product features.

5.6.2.4 Abstractive and Extractive Summarization

Our approach is extractive based summarization. We compared our method with the state of the art abstractive summarization [21], which is an unsupervised optimization approach. to see if there is any clear preference for either of the methods. We had a user study of 5 graduate students in our lab. We show 24 products randomly selected from CNET dataset. For each product we provided 2 lines of extractive summary and 4-5 lines of abstracted summary from [21]. For all the 24 products, we asked our subjects which of the two sets they find more informative. As shown in Figure 5.9, the number of the questions preferred by human subjects is almost the same for both methods, meaning that there are some products that abstract summarization is more informative and some others that extractive works better. In total we had 8 products that were preferred equally (50%). In this dissertation, we focus on extracting a subset of short text content that can be served as representation of the whole input text rather than creating a concise, rephrased summarization.

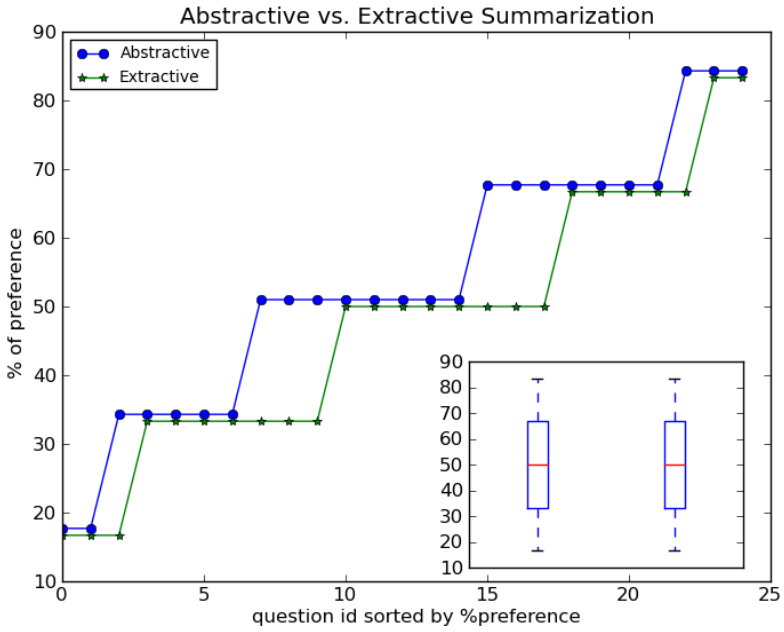

Figure 5.9: The number of the questions preferred by human subjects is almost the same for both methods.

5.6.3 Experiments with Human Evaluation

All the experiments in this section is applied to the YouTube dataset. Section 5.6.3.1 and 5.6.3.2 are applied on all the 17,600 available videos and the rest of this section is based on human curation on 30 videos.

5.6.3.1 Comparing Clustering Methods for Short Text Data

First, we would like to know if clustering the short text, improves the task of summarization. For the final goal of summarization, the intuition is that constructing clusters of similar short text data would result in diversity of content in the summary. To validate this statement we compare the NDCG of the ranking results for PageRank based method and its combination with LDA based clustering method. Figure 5.10 shows that the combination of LDA based clustering with PageRank-based method improves the results.

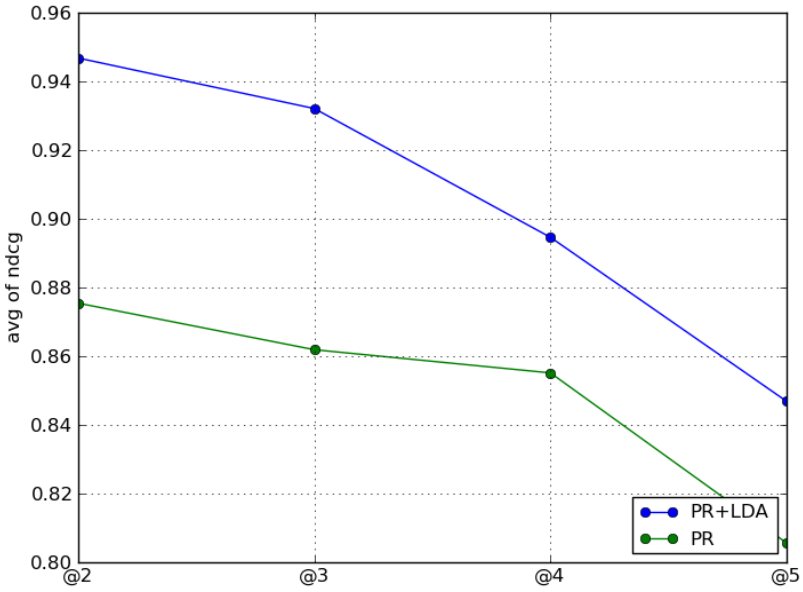

Figure 5.10: PageRank-based method with and without LDA clustering.

Also we would like to select a suitable cluster number. Figure 5.11 shows that in topic based clustering the the ranking results are robust for cluster numbers of 2,3, and 4. We used 3 as the cluster number for the rest of our experiments.

Next we would like to compare the two well-known clustering methods explained earlier to see which one is more suitable for the short text data. Measures of cluster quality are *Cohesion* and *Separation*. The *Cohesion* measures how similar the short text data in one cluster are to each other:

$$coh(X) = \sum_{c_i, c_j \epsilon X} sim(c_i, c_j)$$

On the other hand, the *Separation* measures how dissimilar are the short text data

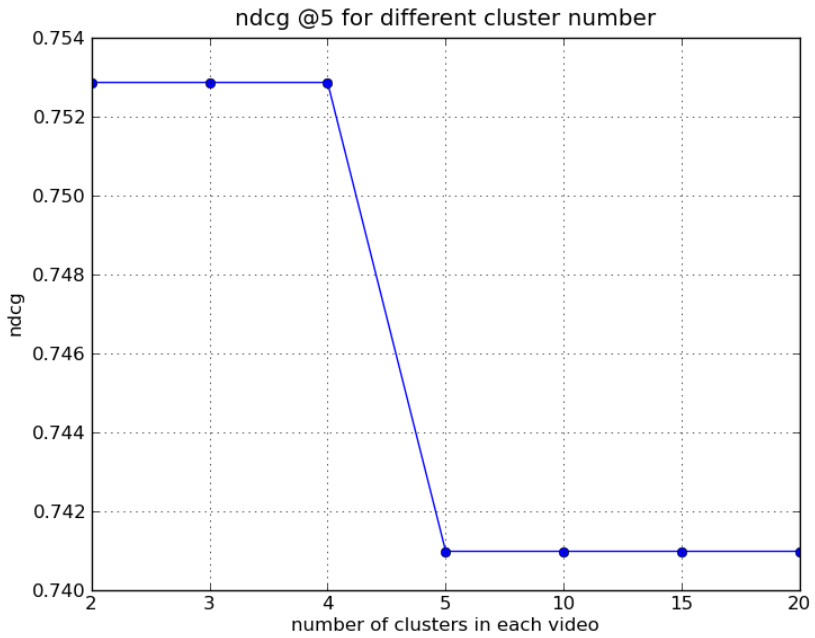

Figure 5.11: NDCG for different cluster numbers in the topic based clustering method.

across different clusters:

$$sep(X,Y) = \sum_{Y \neq X} \frac{1}{\sum_{c_i \epsilon X} sim(c_i, \mu_Y)}$$

The similarity metric we used is cosine similarity which is based on the vector space model of each short text data.

$$sim(c_i, c_j) = \cos(\theta) = \frac{c_i \cdot c_j}{|c_i||c_j|}$$

We apply these two measures to the clusters of comments in YouTube dataset. We also study if considering a specific part of speech (POS) will improve the cluster quality.

Figure 5.12 shows that the Part of Speech (POS) distinction does not help to have

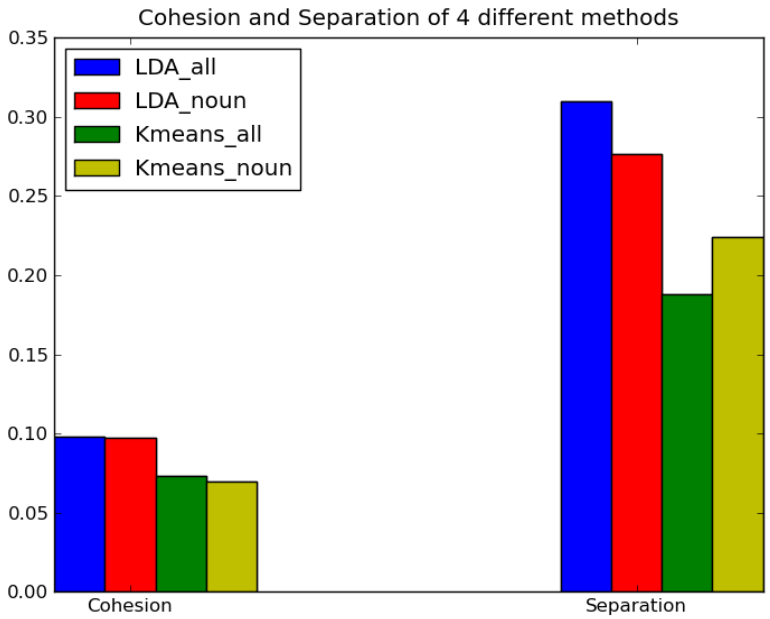

Figure 5.12: Cohesion and Separation of K-means and topic based clustering with their variations: LDA_all and K-means_all use all the terms in a comment, LDA_noun and K-means_noun use only nouns in a comment.

higher quality clusters. This result can be justified by the small number of terms available for comments in which data will be lost by eliminating some terms based on their POS. We also can conclude that the topic based clustering method gives us higher separation and cohesion in comparison with k-means clustering algorithm. Therefore, we use topic based clustering for grouping thematically-related groups of short text.

5.6.3.2 Comparing Discriminative Power of Video Fields

As a part of this study, we want to understand which of the four information fields (tile, keywords, description and comment) encodes more information about a video. That is which field retrieves more specific results in compare with others. If we have a field with restricted vocabulary, then it will have only a limited ability to

distinguish one video from another. As an example, suppose that most of the videos contain the same phrase "game" in their description fields. When users search for the term "game" in the query, most of the videos will be retrieved as the result of search. But, there are few videos that contain the phrase "call of duty" in their meta data. Therefore, if one search for this phrase, the returning results would be more specific. We say that this phrase has a *discriminative power* on identifying the desirable videos. To understand the nature of these fields, we compare the probability distribution of the terms in one video to the probability distribution of the terms in all the videos for each field. If the term distribution in one video is different from the term distribution in all the videos for one specific field, we say that the field has a good *discriminative power*. To measure the discriminative power of each field we use *KL-divergence*.This metric has been used to estimate the potential ability of the terms to improve the search accuracy [98]. We use this metric to find how much a field can make discrimination between a video and all other videos. The Kullback-Leibler divergence between two distributions is specified over the M variable values in vector X. M is the total number of the terms in our corpus dictionary. P is length-M probability distribution of terms in a field for one video; $P = [x_1, x_2, ..., x_M]$, and Q is a length-M probability distribution of terms in a field of all of the videos; $Q = [x_1, x_2, ..., x_M]$. Here x_i is $tf\text{-}idf$ value of the $term_i$ in each distribution.

$$KL(P(x)||Q(x)) = \sum_{x \in Dic} P(x) \log \frac{P(x)}{Q(x)} \tag{5.16}$$

Like what [98] concluded we also found that considering all of the terms in comments gives us lower KL-divergence. This is because of the noise that we encounter in the huge amount of data appeared in comments. However, considering the top 10 terms based on $tf\text{-}idf$ in each model, we found that the comments are more

informative than other fields. Figure 5.13 shows the histogram for KL-Divergence of title, keywords, description and comments. We see that the number of videos with higher KL-Divergence for the comment field is much more than the title, description and keyword fields.

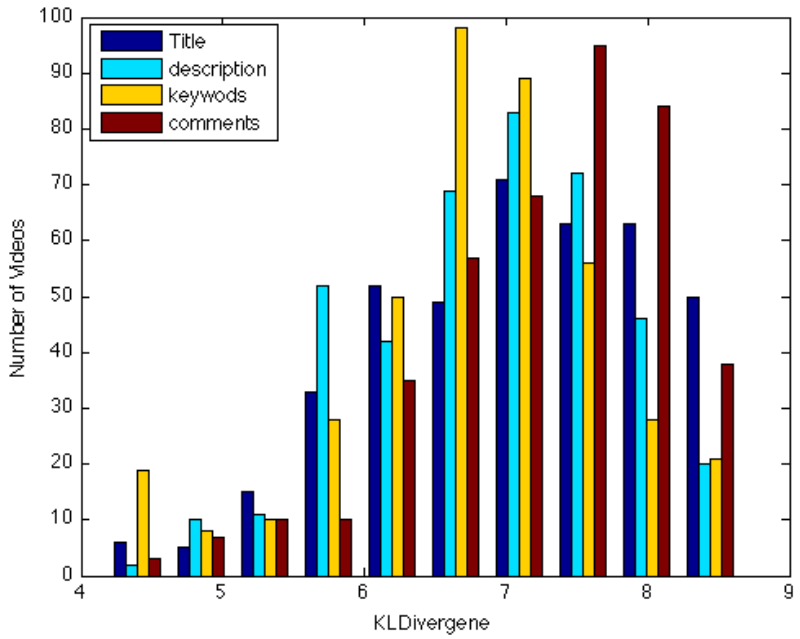

Figure 5.13: KL-Divergence for different fields of YouTube videos. Higher KL-divergence means more discriminative power.

5.6.3.3 Comparing Different Graph Based Ranking Algorithms

Among different graph based ranking algorithms, Mead and LexRank methods have shown good results for single or multi documents summarization when provided with some pages of concrete articles. The position of the sentences and the similarity of them to the title of resource are among strong features that are used in these methods. In the short text ranking problem we also would like to pick up the sentences that are playing the role of summary or abstract of all of the input short text data. Therefore, we look at the fist few short text in our ranking results as the

summary of all input short text. Figure 5.14 shows that the proposed PageRank based method works better on the YouTube comments. We can not consider any of the comments as a leader similar to the sentences in the abstract of an article in traditional summarization. Therefore, the traditional methods are not suitable for ranking the comments of online community. Three default features that come with the MEAD distribution are Centroid, Position, and Length. Position is the normalized value of the position of a sentence in the document such that the first sentence of a document gets the maximum Position value of 1, and the last sentence gets the value 0. In our case we have ignored this feature by assigning weight zero to it. Because, in the comments we do not have the logical orders that we see for example in a piece of news. Length is not a real feature score, but a cut off value that ignores sentences shorter than the given threshold. Here we considered the threshold length of 2 and 5 and found very similar results.

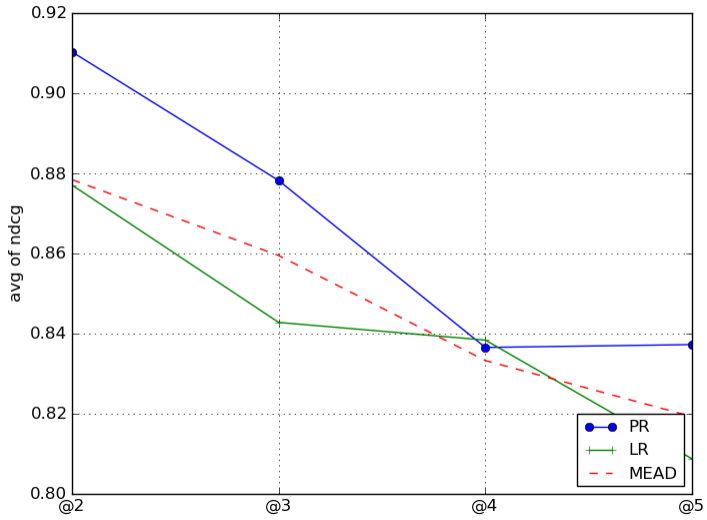

Figure 5.14: Compare PageRank with available graph based methods (Mead and LexRank). Higher NDCG is more desirable.

5.6.3.4 *In-cluster Ranking Evaluation*

Having the clusters of short text, we would like to study which of the in-cluster ranking method will be more effective. Figure 5.15 compares the information theory-based method (MI) with vector space (geometric) based importance (tf-idf) when it is applied inside each of the result clusters. It suggests that MI offers better performance than tf-idf in selecting representative comments for the first few selected comments. This can be justified by MI's focus on terms that contribute the most to a particular cluster. However, calculating the MI of each term in each cluster is computationally expensive.

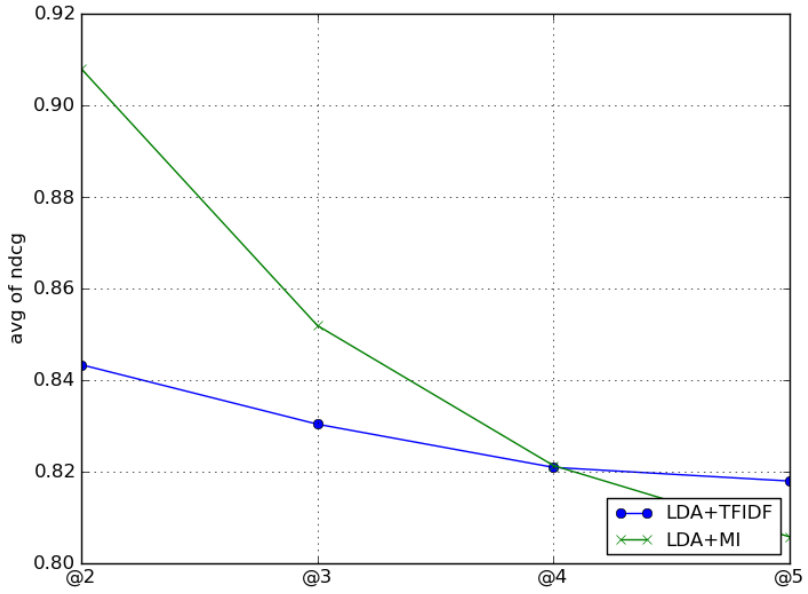

Figure 5.15: Compare the tf-idf and MI based in-cluster ranking

Here we study different versions of the PageRank algorithms to find the most optimized parameters for the experiments. We first change the threshold parameter p in which determines how many common terms between two comments are needed

to link them in the graph. On the other hand, we are interested in knowing that if flipping the direction of the links would improve the results. Figure 5.16 shows that we have the highest performance when we use $p = 3$ as a threshold to connect two comments together and the forward linking from newer to older comments gives better results than backward direction.

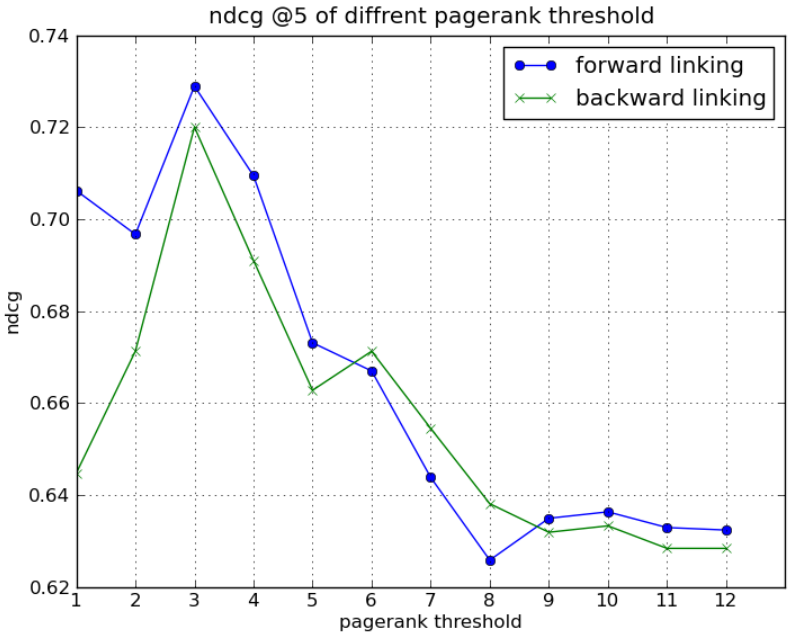

Figure 5.16: Compare thresholds and directions for PageRank based algorithm.

We also consider other similarity measures among the comments, i.e, instead of linking comments based on certain amount of common terms, we consider the Jaccard coefficient of the two comments. The results has shown that using Jaccard similarity measure as an alternative does not improve the results.

5.6.3.5 Combination of different methods

Comparing different combinations of the proposed algorithms, we found that PR (PageRank) and PR+LDA are more successful than LR (LexRank) and LR+LDA.

See Figure 5.17. Also adding topic based clustering, LDA, to each of these methods improves the results.

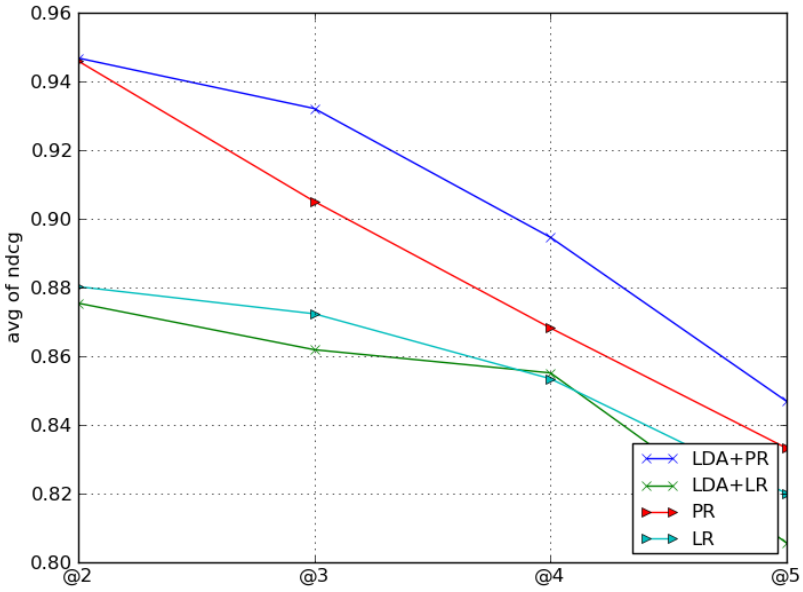

Figure 5.17: Comparing PageRank with LexRank and their combinations with LDA clustering.

Figure 5.18 shows that the combination of two families of approaches, such as cluster based (LDA) vs. ranking based (PR) methods give better results than individual methods. The reason is, each of the topic based clustering and PageRank based ranking are focusing more on one important aspect of prioritizing comments. One weights the thematically coherence of the comments in its cluster and the other one is focused on the term usage of the earlier comments and how much it was attracted by the later users. In this figure we used MI as in-cluster ranking method. LDA-MI means that we first cluster the similar short text into topics and then we sort each cluster based on the MI value.

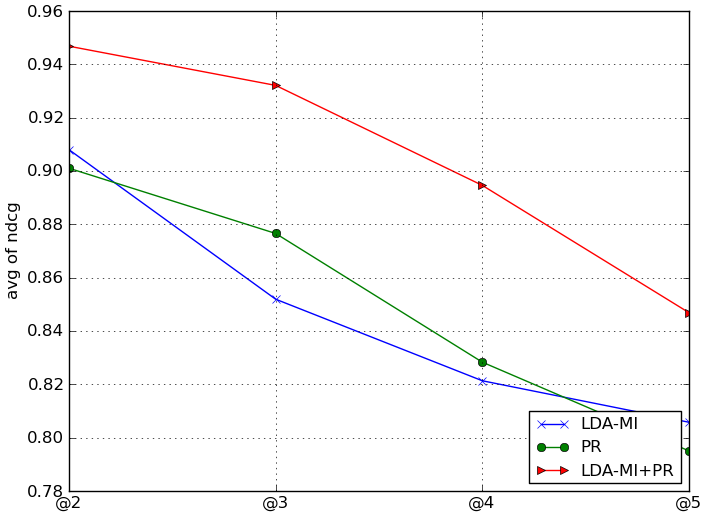

Figure 5.18: Comparing PR, $LDA - MI$ and their combination.

Finally Figure 5.19 shows all the different methods in compare with the random method.

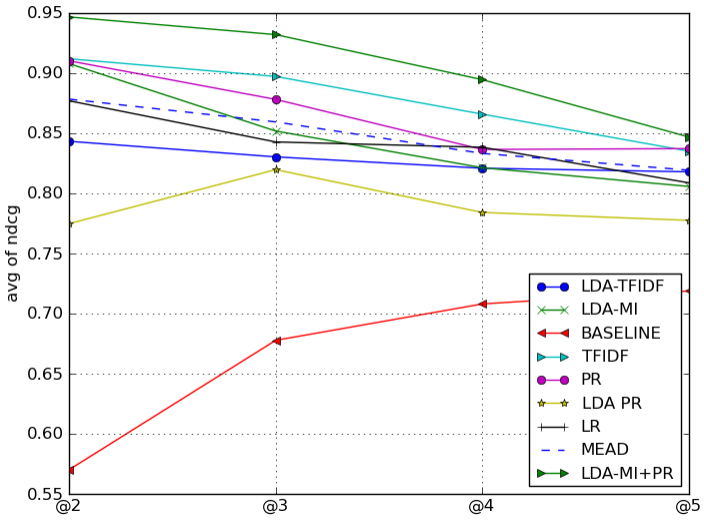

Figure 5.19: Compare all the proposed methods vs. random.

5.6.4 Experiments with Automatic Evaluation

Now that we obtain intuition behind different methods of summarization, we would like to apply automatic evaluation on large datasets. We have used the 3 datasets mentioned before to see the effect of each of the summarization variations on the fully automatic metrics, that is KL-divergence and Retention Rate.

5.6.4.1 Amazon Dataset Automatic Evaluation

Here we compare different versions of PageRank based algorithms (PR, WPR, PRT, WPRT) on the Amazon dataset. Figure 5.20 shows that considering the edge weight and *tf-idf* of the common terms are not improving the proposed PageRank algorithm. So we do the rest of the experiments with the the original version of proposed PageRank algorithm (PR).

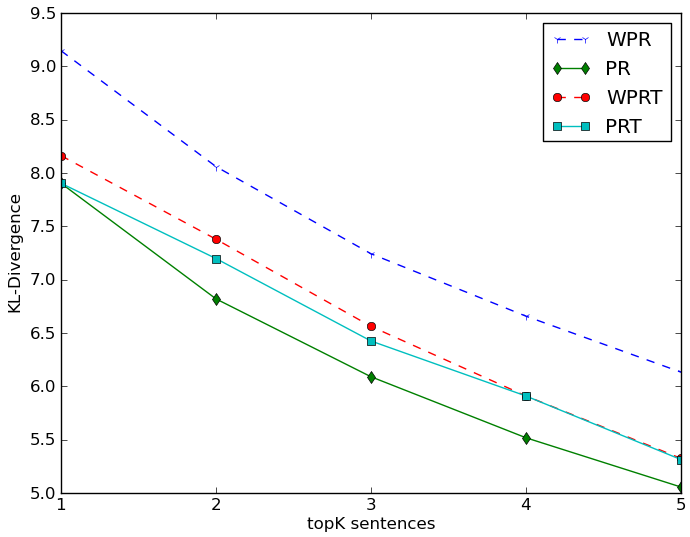

Figure 5.20: Amazon Dataset: Comparison of different versions of PR algorithm. Lower KL-divergence is desirable.

We also compare the Retention Rate (RR) of these methods. Fixing the length of our summary, higher retention rate is what we would like to achieve. See Figure

5.21. We do not see any difference among different versions of PR algorithm based on Retention Rates, However we see slightly better results when considering edge weights in compare with considering the tf-idf of the common terms.

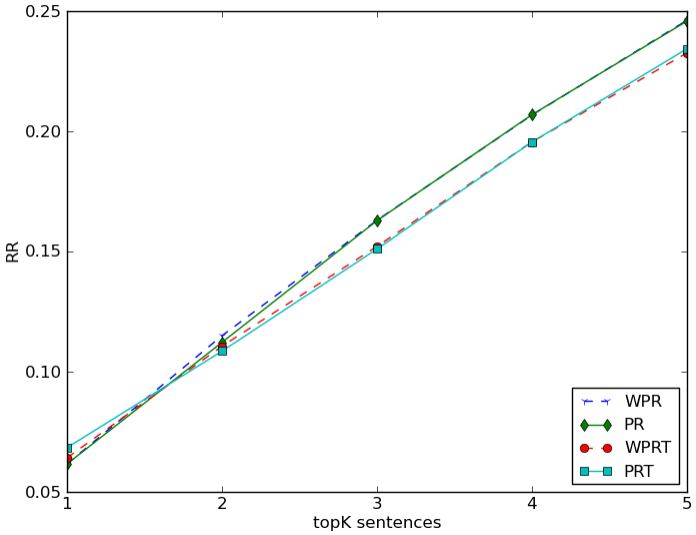

Figure 5.21: Amazon Dataset: Comparison of different versions of PR algorithm. Higher RR is desirable.

Comparing PageRank based ranking (PR) with TFIDF based ranking, their combinations with LDA clustering and the highlights of Amazon in Figure 5.22, we see that PR method has the lowest KL-divergence and therefore the most favorable method. Amazon only provides three sentences as the summary. This is why we do not see further decrease in KL-divergence as we increase the number of sentences. Comparing only the fist 3 sentences of the Amazon method with PR method we see PR gives a much better result. Note that adding LDA clustering to each of the proposed methods of PR and TFIDF does not lower the KL-divergence. Figure 5.23 shows the whisker plot for the same methods. In this plot we can visualize the distribution of data more clearly. Figure 5.24 compares the Retention Rate of these methods. We find that topic clustering method in combination with TFIDF will give

the Retention Rate higher than Amazon. PR gives similar results to Amazon and the rest are not as good as Amazon summary.

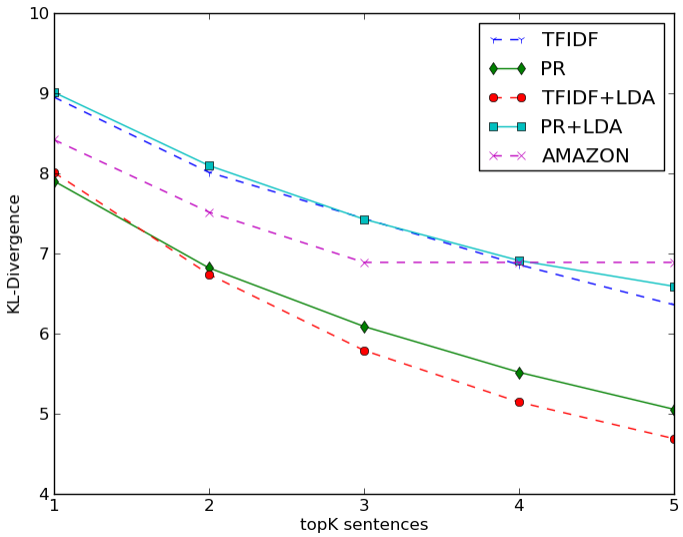

Figure 5.22: Amazon Dataset: Comparison of PR and TFIDF methods with their LDA versions.

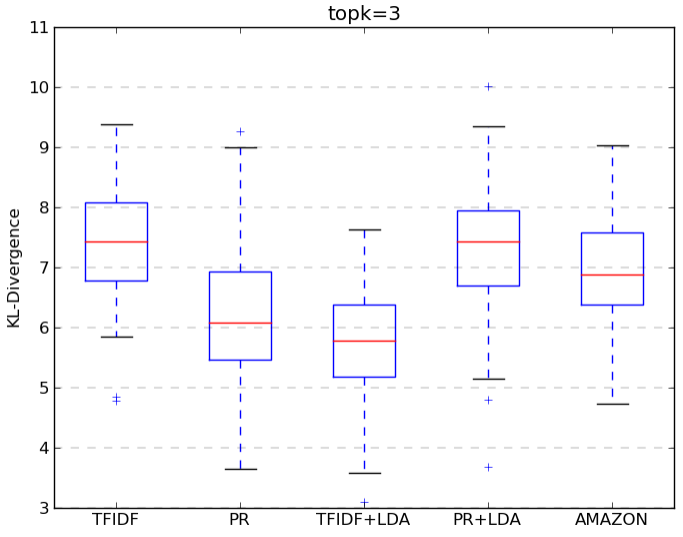

Figure 5.23: Amazon Dataset: Whisker Plot for Comparison of PR and TFIDF methods with their LDA versions.

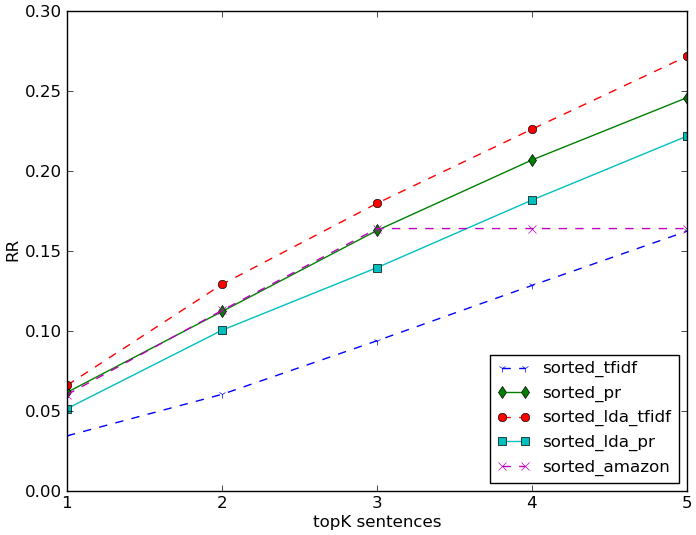

Figure 5.24: Amazon Dataset: Comparison of Retention Rate for PR and TFIDF methods with their LDA versions.

5.6.4.2 CNET Dataset Automatic Evaluation

Here we compare different versions of PageRank based algorithms (PR, WPR, PRT, WPRT) on the CNET dataset. Figure 5.25 shows that WPR and PR methods gives lower Kl-divergence than WRT and PRT and therefore are more desirable. Also they both show higher Retention Rate in Figure 5.26. These two Figures show the reason behind our preference of PR and WPR over those versions with considering tf-idf.

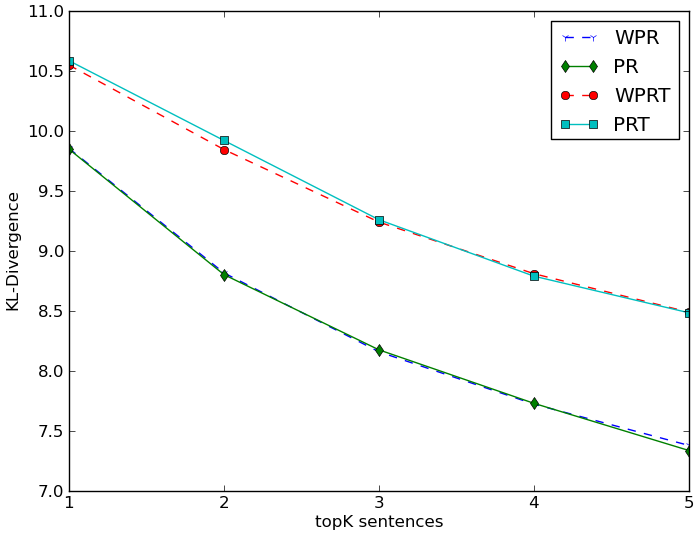

Figure 5.25: CNET Dataset: Comparison of different versions of PR algorithm. Lower KL-divergence is desirable.

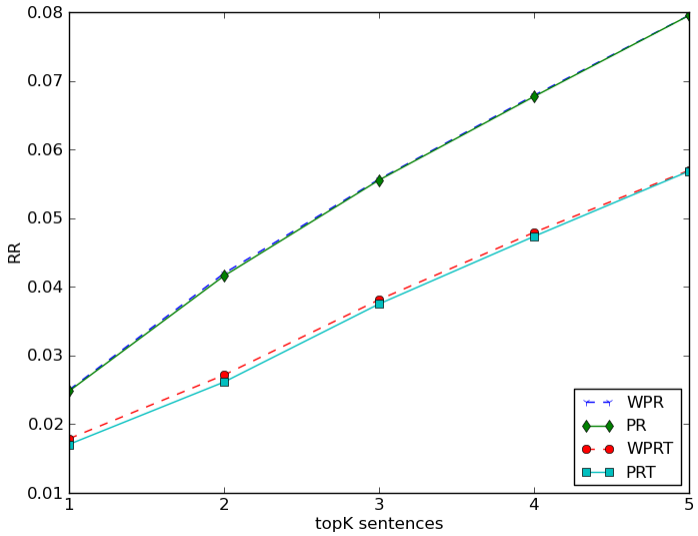

Figure 5.26: CNET Dataset: Comparison of different versions of PR algorithm. Higher RR is desirable.

Now we compare the ranking based on PR and TFIDF and their combinations with LDA. Figure 5.27 and its corresponding whisker plot in Figure 5.28 show that

PR is better than all other methods. On the other hand, adding LDA will improve the ranking based on TFIDF but not the ranking based on PR. Figure 5.29 compares the Retention Rate of these methods. We find that TFIDF alone and PR alone gives highest Retention Rate in compare to others.

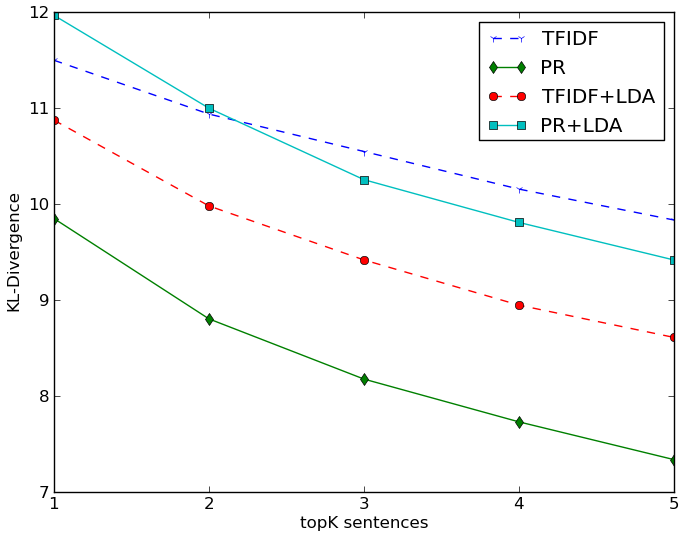

Figure 5.27: CNET Dataset: Comparison of PR and TFIDF methods with their LDA versions.

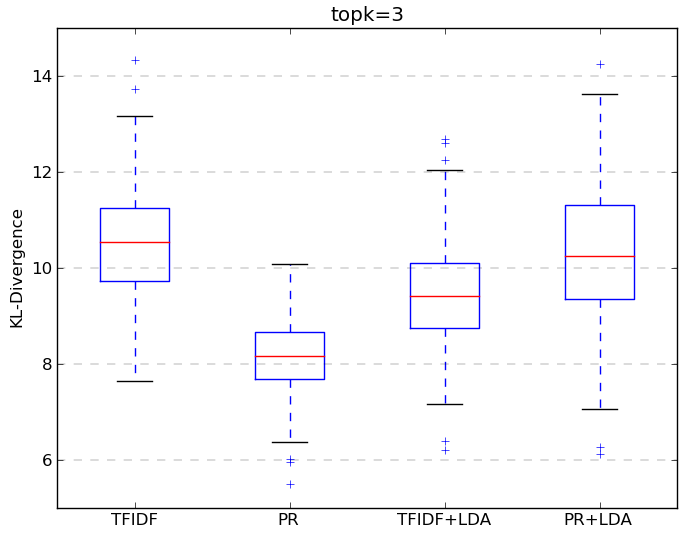

Figure 5.28: CNET Dataset: Whisker Plot for Comparison of PR and TFIDF methods with their LDA versions.

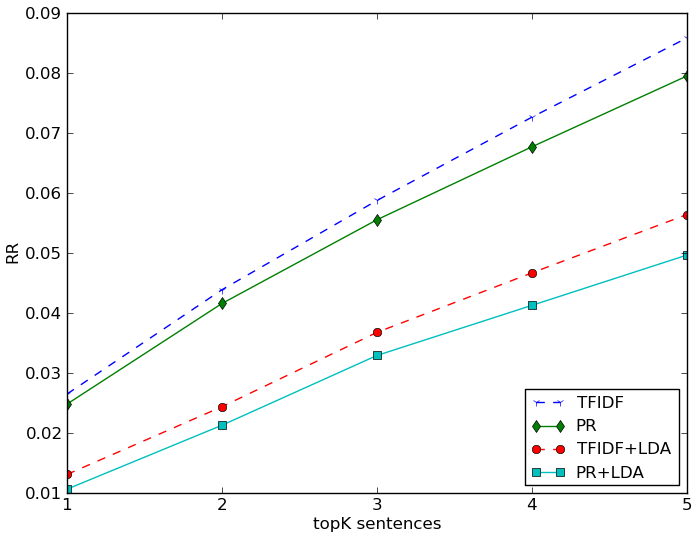

Figure 5.29: CNET Dataset: Comparison of Retention Rate for PR and TFIDF methods with their LDA versions.

5.6.4.3 YouTube Dataset Automatic Evaluation

Finally we compare different versions of PageRank based algorithms (PR, WPR, PRT, WPRT) on the YouTube dataset. Figure 5.30 and 5.31 shows very similar KL-divergence for all the PR versions and a slightly better Retention Rate for PR and WPR in compare with PRT and WPRT.

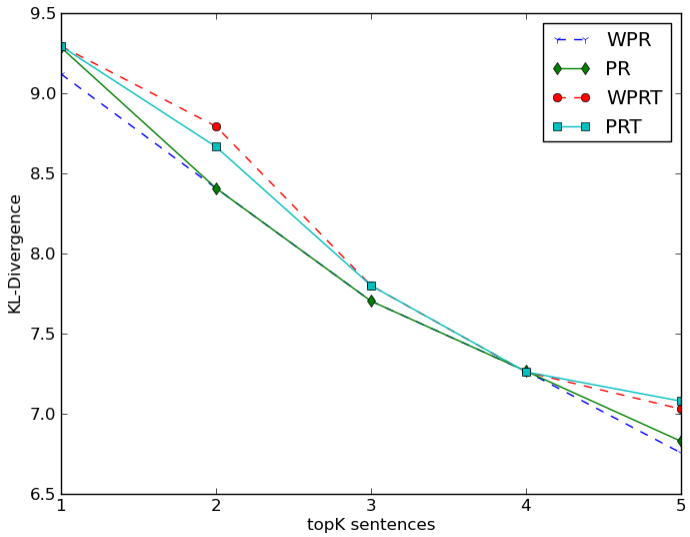

Figure 5.30: YouTube Dataset: Comparison of different versions of PR algorithm. Lower KL-divergence is desirable.

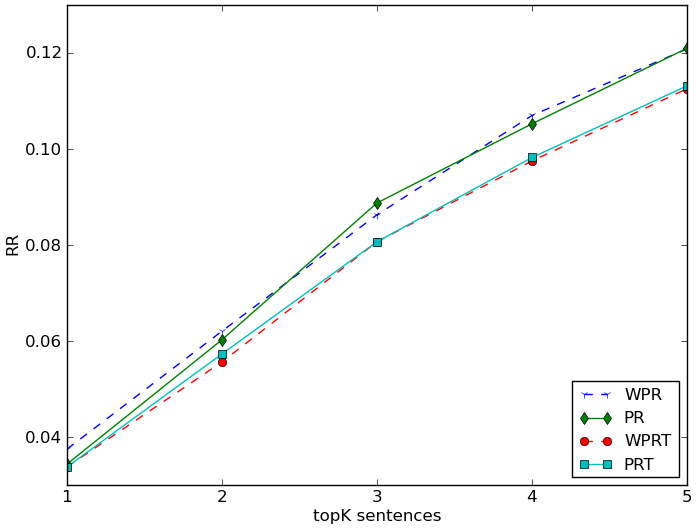

Figure 5.31: YouTube Dataset: Comparison of different versions of PR algorithm. Higher RR is desirable.

Now we compare ranking based on PR, TFIDF and their combinations with LDA. Figure 5.32 and its corresponding whisker plot in Figure 5.33 show that combination of TFIDF and LDA will give a the lowest KL-divergence. In here adding LDA to TFIDF has improved the TFIDF alone significantly. Figure 5.34 compares the Retention Rate of these methods. We find that topic based clustering method in combination with TFIDF will give the highest Retention Rate. PR is the next method with highest Retention Rate.

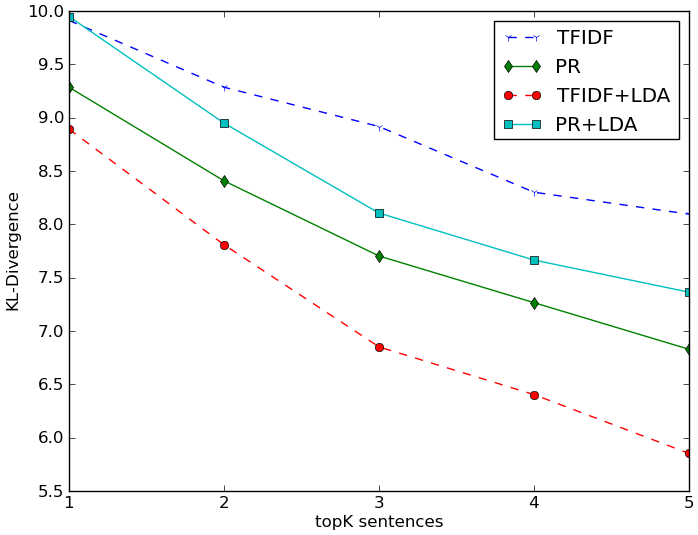

Figure 5.32: YouTube Dataset: Comparison of PR and TFIDF methods with their LDA versions.

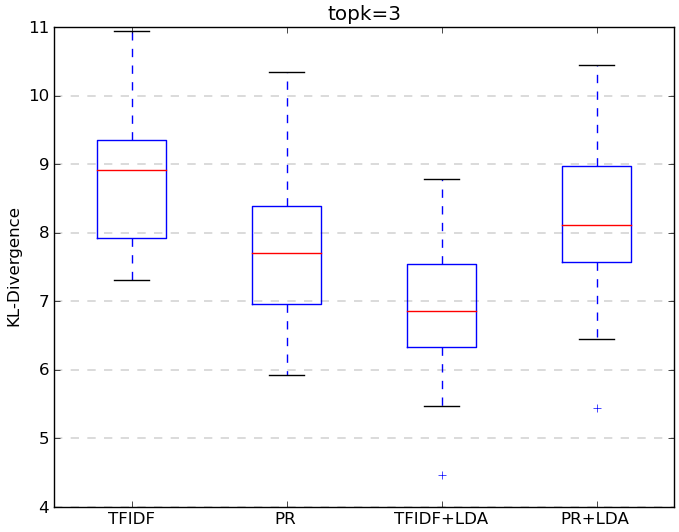

Figure 5.33: YouTube Dataset: Whisker Plot for Comparison of PR and TFIDF methods with their LDA versions.

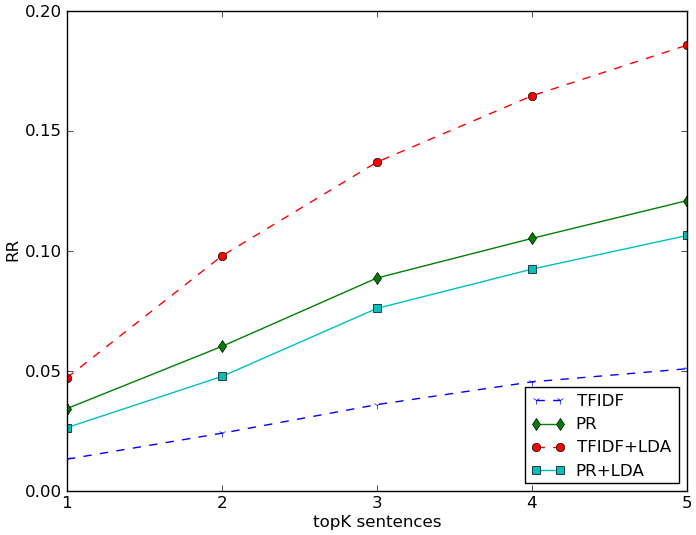

Figure 5.34: YouTube Dataset: Comparison of Retention Rate for PR and TFIDF methods with their LDA versions.

5.7 Conclusion

In this chapter we studied different methods of extractive summarization. We applied human evaluation for small number of data and automatic evaluation metrics on a large amount of data. For human evaluation, we found that topic based clustering is a more suitable clustering method for short text data in compare with the well-known K-means clustering algorithm. On the other hand, combination of topic based clustering and PageRank based ranking shows better results. For large scale automatic evaluation, we found that adding clustering to TFIDF based ranking improves the results significantly. However, in 2 out of 3 large dataset we found that PageRank based ranking over the comments of a resource provides lowest KL-divergence which is desirable.

6. CONCLUSION

In this dissertation, we have introduced and developed algorithms and techniques to promote high quality content, organize the short text, and summarize them effectively and efficiently. There are three objectives that we have discussed throughout this dissertation:

First, we addressed methods to automatically rank the comments associated with a social web object based on the expressed preferences of the community itself. By learning ranking functions for user-contributed comments, we provided a sound basis for enhanced comment-based social web applications like summarization and content retrieval. We proposed and evaluated a regression-based learning model for automatically identifying comment quality within a Social Web community based on the community's preferences. We examined the impact of different comment features like visibility, user reputation of the comment's author, and the content of the comment itself to understand the influence of these features on the overall community's preference for comments.

Measuring the classification rate, precision, and recall over the test set of comments, shows that both linear regression and quadratic classifier approaches have high classification rate as well as high precision and recall in most groups. For example, the precision to identify the fair and good comments reaches to 82% and 70%. For the ranking problem the goal is not to precisely estimate the actual comment community rating for a comment. Instead, the *relative order* of comments needs to be predicted, so that even as new ratings are made on the comments, the model will be able to capture the relative quality.

Second, we studied the problem of automatically assigning labels to short text

in social media. We proposed a graph-based prediction framework for increasing the coverage of semantic annotations in real-time web status updates. We saw how the path aggregation technique for scoring the closeness of terms and hashtags in the graph, pivot term selection, and the dynamic sliding window led to encouraging results in comparison with alternative methods. In this way, the feedback between small-scale curation and automated methods may provide an evolving framework for ongoing organization of real-time web content.

Association Rule showed the most promising results among the other alternative methods. Comparing the proposed semantic based tag recommendation method with the state of the art association rule method shows a much higher recall for the proposed method. The recall for association rule is 5%, 5%, 10%, 25% for hourly, four hours, one day and one week training sliding window respectively where these numbers are 39%, 35%, 40%,42% for the proposed method. Measuring the precision, there is not much of a difference between the two methods. However having a larger window will improve the precision and recall for both of them.

Third, we explored the summarization approaches on the short text in a way that it reflects diverse viewpoints of different users, while retaining the key concepts with high text quality in the summary. We proposed a general approach and evaluated this approach over a collection of YouTube videos and product reviews. Our first hypothesis was if we first cluster the comments of each resource and then select the most informative comment from each cluster, we get a good set of representative sentences. Among K-means and topic based clustering, topic-based clustering gave us better results. For comment selection, the term-importance based ranking were examined and both showed better results than baseline methods, with the mutual information approach showing the most success.

Comparing the PageRank based ranking approach with traditional document

summarization approaches such as LexRank, MEAD shows that PageRank could result in a better performance. NDCG of the PageRank is 91% for the first two selected sentences whereas the NDCG of the MEAD and LexRank methods are both almost 88%. Also we showed that adding topic based clustering method such as LDA to the PageRank based method will result in a higher NDCG 94% vs. 87%.

Exploring appropriate methods to organize, distill and summarize of the social media websites was the main rationale to conduct this research. We were able to identify preferred, high quality short text from a massive amount of data mentioned by millions of users with variable style and quality. We were as well be able to organize and label such short text data in a real-time manner, in addition to constructing a summarizing framework to capture the gist of such content. To apply the algorithms and techniques developed in this dissertation, we have collected large datasets of Digg, Twitter, Youtube and Amazon with more than millions of users and hundred millions of generated content.

6.1 Future Work

As part of our future work, we are interested to integrate these results as part of our broader research effort to build enhanced Social Web information management applications that leverage this social collective intelligence. We are also interested to augment the baseline model presented here with information from each user's social network, so that hashtags adopted by a user's community may provide a more personalized set of hashtag recommendations. We are also interested to study the impact of increasing spam and low-quality hashtags on the performance of hashtag prediction. Knowing who is writing a review and which location he is from, would help to have a more customized summaries for different users. So if user A is leaving in the same location as user B, his concerns about "car tire" would be similar. The

same goes with the friendship relationship. Some users might like to consider what their friends have mentioned about some features of a product. In this dissertation we did not cover these important aspects. It could be continued for the future. There are different methods to build a network of users. Here are some examples when we use a social network such as Twitter:

- A social network with friendship relationships. Examples include a network based on the *mentions* in Twitter, where we have a network of friends who communicate densely with each other and use @ to directly talk to each other. Also when we construct the network based on follower, followee relationships, we have a network based on the friendships.

- A network of users with similar tastes that do not know each others necessarily. For example, when we filter the tweets about specific hashtags or topics, then we have a community of users with similar interests who are talking about a specific topic. Having a network based on the re-tweets happening in twitter will also highlight the users interested in particular topics.

- A network of friends that have similar tastes. This is a combination of the past two networks.

It will be interesting to identify power users in such networks. The three types of features to identify power users can be as the following.

Network based Features: There are multiple centrality measures that will help us to understand how important a user as a node is in a network graph. Now that we have a network constructed by one of the previous proposed methods, we need to find the nodes that are more central to all other ones. Our purpose is to give higher scores to the content mentioned by central users. In addition such users would be

good choices of receiving clearance coupons, free samples, and any other types of targeted advertisement services. Some of the centrality measures for a single node A are:

- $closeness(A)$: It is the reverse of the average distance of A to all other nodes. It shows how much close a node is to all other nodes.
- $betweenness(A)$: It is the summation of the fraction of all-pairs shortest paths that pass through node A. It shows how much a node is between other nodes.
- $centrality(A)$: Which is the Fraction of nodes that A is connected to.
- $degree(A)$: The degree of a node A in the network. It means the number of neighbors a node has.

Review based Features: One method to identify power users is to consider the user-generated meta-data (comments, tweets, reviews) of such users in the social websites. If their contributed text is found to be useful then they are considered as influential users. Now the question is how to identify the usefulness of the text generated by such users. There are lots of explicit and implicit features available for this purpose. Explicit features includes the number of re-tweets, votes, helpful remarks from the web community. Implicit features include the text score which is received by algorithms that can identify high quality text. Such algorithm take into account NLP type of features, entropy of the text and the specificity of the term to the category it belongs to.

Purchase-History based Features: Another method is to find users that are *experts* in a specific category. For example those users that have purchased many of the video games such as "Call of Duty" in the past, may have a better influence on

those users that are novice to the area. Their comments and reviews should also be highlighted more than the normal users.

Spread-History Features: To identify influential users one method is to probe in the user history to see how *deep* a single behavior of a user has caused the rest of the network to follow the same pattern of behavior. Concretely if a user has purchased a product at time t_1, how many of the immediate friends, and how many of the friends of immediate friends have purchased the same product at the time t_2 in which $t1 < t2$.

By identification of influential users effectively and highlighting their distributed content with high quality we are creating a framework which answers the existing need for an automatically organized, distilled and categorized social media for the web community.

6.2 Final Thoughts

The expected outcome of this research would be effective algorithms to rank, label and summarize the short text content in social websites. By exploring how a community can self-regulate, we may gain insights into what the community values are and how to sustain the positive growth of the community. This effort could be integrated to build enhanced social web information management applications that leverage such social collective intelligence. Having a framework of learning the context of the short-text and recommend relevant labels would be beneficial to extend the small fraction of self-curated messages to organize the vast majority of messages that have not been annotated. As systems like Twitter and Facebook continue to grow, the proposed labeling approach could be used to extend the small fraction of self-curated messages to organize the vast majority of messages that have not been annotated. In summary, this thesis would benefit many social websites including

those who manage users' status updates (Twitter, Facebook, LinkedIn), those who deal with user-contributed comments (news websites, weblogs, forums), and the e-commerce websites who handle large scale customer reviews to better organize, rank, and summarize their content for the web community.

REFERENCES

[1] AFP. Googlenews. 24 hours of video uploaded to youtube every minute, March 2010.

[2] Eugene Agichtein, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. Finding high-quality content in social media. In *Proceedings of the international conference on Web search and web data mining*, pages 183–194. ACM, 2008.

[3] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. 22(2):207–216, 1993.

[4] James Allan, Courtney Wade, and Alvaro Bolivar. Retrieval and novelty detection at the sentence level. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 314–321. ACM, 2003.

[5] R. Barzilay, M. Elhadad, et al. Using lexical chains for text summarization. In *Proceedings of the ACL workshop on intelligent scalable text summarization*, volume 17, pages 10–17, 1997.

[6] P. B. Baxendale. Man-made index for technical literature - an experiment. *IBM Journal of Research and Development*, 2(4):354–361, 1958.

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[8] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In

Annual Meeting-Association For Computational Linguistics, volume 45, page 440, 2007.

[9] Andrew Byde, Hui Wan, and Steve Cayzer. Personalized tag recommendations via tagging and content-based similarity metrics. In *Proceedings of the International Conference on Weblogs and Social Media*, 2007.

[10] Bob Carpenter. Phrasal queries with lingpipe and lucene: ad hoc genomics text retrieval. In *Proceedings of the 13th annual text retrieval conference*, 2004.

[11] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[12] C. Chen, F. Ibekwe-SanJuan, E. SanJuan, and C. Weaver. Visual analysis of conflicting opinions. In *Visual Analytics Science And Technology, 2006 IEEE Symposium On*, pages 59–66. IEEE, 2006.

[13] Xu Cheng, Cameron Dale, and Jiangchuan Liu. Understanding the characteristics of internet short video sharing: Youtube as a case study. *arXiv preprint arXiv:0707.3670*, 2007.

[14] Paul-Alexandru Chirita, Stefania Costache, Wolfgang Nejdl, and Siegfried Handschuh. P-tag: large scale automatic generation of personalized annotation tags for the web. In *Proceedings of the 16th international conference on World Wide Web*, pages 845–854. ACM, 2007.

[15] John M Conroy and Dianne P O'leary. Text summarization via hidden markov models. In *Proceedings of the 24th annual international ACM SIGIR conference*

on Research and development in information retrieval, pages 406–407. ACM, 2001.

[16] W Bruce Croft, Donald Metzler, and Trevor Strohman. *Search engines: Information retrieval in practice*. Addison-Wesley, 2010.

[17] Harris Drucker, Chris JC Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. pages 155–161. Morgan Kaufmann Publishers, 1997.

[18] H.P. Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969.

[19] Günes Erkan and Dragomir R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res. (JAIR)*, 22:457–479, 2004.

[20] K. Ganesan, C.X. Zhai, and J. Han. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 340–348. Association for Computational Linguistics, 2010.

[21] Kavita Ganesan, ChengXiang Zhai, and Evelyne Viegas. Micropinion generation: An unsupervised approach to generating ultra-concise summaries of opinions. In *Proceedings of the 21st international conference on World Wide Web*, pages 869–878. ACM, 2012.

[22] Kavita Ganesan, ChengXiang Zhai, and Evelyne Viegas. Micropinion generation: An unsupervised approach to generating ultra-concise summaries of

opinions. In *Proceedings of the 21st international conference on World Wide Web*, pages 869–878. ACM, 2012.

[23] Nikhil Garg and Ingmar Weber. Personalized, interactive tag recommendation for flickr. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 67–74. ACM, 2008.

[24] Mark Girolami and Ata Kabán. On an equivalence between plsi and lda. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 433–434. ACM, 2003.

[25] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[26] Scott Golder and Bernardo A Huberman. The structure of collaborative tagging systems. *arXiv preprint cs/0508082*, 2005.

[27] Ziyu Guan, Jiajun Bu, Qiaozhu Mei, Chun Chen, and Can Wang. Personalized tag recommendation using graph-based ranking on multi-type interrelated objects. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 540–547. ACM, 2009.

[28] Yusef Hassan-Montero and Víctor Herrero-Solana. Improving tag-clouds as visual information retrieval interfaces. In *International Conference on Multidisciplinary Information Sciences and Technologies*, pages 25–28. Citeseer, 2006.

[29] Paul Heymann, Daniel Ramage, and Hector Garcia-Molina. Social tag prediction. In *Proceedings of the 31st annual international ACM SIGIR conference*

on Research and development in information retrieval, pages 531–538. ACM, 2008.

[30] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[31] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[32] E. Hovy and C.Y. Lin. Automated text summarization and the summarist system. In *Proceedings of a workshop on held at Baltimore, Maryland: October 13-15, 1998*, pages 197–214. Association for Computational Linguistics, 1998.

[33] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.

[34] M. Hu and B. Liu. Mining opinion features in customer reviews. In *Proceedings of the National Conference on Artificial Intelligence*, pages 755–760. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

[35] Meishan Hu, Aixin Sun, and Ee-Peng Lim. Comments-oriented document summarization: understanding documents with readers' feedback. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 291–298. ACM, 2008.

[36] Jens Illig, Andreas Hotho, Robert Jäschke, and Gerd Stumme. A comparison of content-based tag recommendations in folksonomy systems. In *Knowledge Processing and Data Analysis*, pages 136–149. Springer, 2011.

[37] Tereza Iofciu and Gianluca Demartini. Time based tag recommendation using direct and extended users sets. *ECML PKDD Discovery Challenge 209 DC09*, 497:99–107, 2009.

[38] Danesh Irani, Steve Webb, Calton Pu, and Kang Li. Study of trend-stuffing on twitter through text classification. In *Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS)*, 2010.

[39] Robert Jäschke, Leandro Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in social bookmarking systems. *Ai Communications*, 21(4):231–247, 2008.

[40] Nitin Jindal and Bing Liu. Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining*, pages 219–230. ACM, 2008.

[41] H.D. Kim, K. Ganesan, P. Sondhi, and C.X. Zhai. Comprehensive review of opinion summarization. 2011.

[42] H.D. Kim and C.X. Zhai. Generating comparative summaries of contradictory opinions in text. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 385–394. ACM, 2009.

[43] Jon M Kleinberg. Hubs, authorities, and communities. *ACM Computing Surveys (CSUR)*, 31(4es):5, 1999.

[44] Georgia Koutrika, Frans Adjie Effendi, Zolt Gyöngyi, Paul Heymann, Hector Garcia-Molina, et al. Combating spam in tagging systems: An evaluation. *ACM Transactions on the Web (TWEB)*, 2(4):22, 2008.

[45] Georgia Koutrika, Zahra Mohammadi Zadeh, and Hector Garcia-Molina. Coursecloud: summarizing and refining keyword searches over structured data. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pages 1132–1135. ACM, 2009.

[46] Georgia Koutrika, Zahra Mohammadi Zadeh, and Hector Garcia-Molina. Data clouds: summarizing keyword search results over structured data. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pages 391–402. ACM, 2009.

[47] J. Kupiec, J. Pedersen, and F. Chen. A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73. ACM, 1995.

[48] Cliff Lampe and Paul Resnick. Slash (dot) and burn: distributed moderation in a large online conversation space. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 543–550. ACM, 2004.

[49] Cliff AC Lampe, Erik Johnston, and Paul Resnick. Follow the reader: filtering comments on slashdot. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1253–1262. ACM, 2007.

[50] B. Larsen. A trainable summarizer with knowledge acquired from robust nlp techniques. *Advances in Automatic Text Summarization*, page 71, 1999.

[51] Kyumin Lee, James Caverlee, Krishna Y Kamath, and Zhiyuan Cheng. Detecting collective attention spam. In *Proceedings of the 2nd Joint WICOW/AIRWeb Workshop on Web Quality*, pages 48–55. ACM, 2012.

[52] Kristina Lerman. Social networks and social information filtering on digg. 2006.

[53] Kristina Lerman. Social information processing in news aggregation. *Internet Computing, IEEE*, 11(6):16–28, 2007.

[54] Kristina Lerman and Aram Galstyan. Analysis of social voting patterns on digg. In *Proceedings of the first workshop on Online social networks*, pages 7–12. ACM, 2008.

[55] Beibei Li, Shuting Xu, and Jun Zhang. Enhancing clustering blog documents by utilizing author/reader comments. In *Proceedings of the 45th annual southeast regional conference*, pages 94–99. ACM, 2007.

[56] C.Y. Lin and E. Hovy. Identifying topics by position. In *Proceedings of the fifth conference on Applied natural language processing*, pages 283–290. Association for Computational Linguistics, 1997.

[57] Annie Louis and Ani Nenkova. Automatically evaluating content selection in summarization without human models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 306–314. Association for Computational Linguistics, 2009.

[58] Y. Lu and C. Zhai. Opinion integration through semi-supervised topic modeling. In *Proceedings of the 17th international conference on World Wide Web*, pages 121–130. ACM, 2008.

[59] Yu-Ta Lu, Shoou-I Yu, Tsung-Chieh Chang, and Jane Yung-jen Hsu. A content-based method to enhance tag recommendation. In *Proceedings of the 21st international jont conference on Artifical intelligence*, pages 2064–2069. Morgan Kaufmann Publishers Inc., 2009.

[60] Yue Lu, ChengXiang Zhai, and Neel Sundaresan. Rated aspect summarization of short comments. In *Proceedings of the 18th international conference on World wide web*, pages 131–140. ACM, 2009.

[61] Claudio Lucchese, Gleb Skobeltsyn, and Wai Gen Yee. 7th workshop on large-scale distributed systems for information retrieval (lsds-ir'09). In *ACM SIGIR Forum*, volume 43, pages 34–40. ACM, 2009.

[62] H.P. Luhn. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.

[63] I. Mani and E. Bloedorn. Multi-document summarization by graph search and matching. *arXiv preprint cmp-lg/9712004*, 1997.

[64] G Harry McLaughlin. Smog grading: A new readability formula. *Journal of reading*, 12(8):639–646, 1969.

[65] Q. Mei, X. Ling, M. Wondra, H. Su, and C.X. Zhai. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of the 16th international conference on World Wide Web*, pages 171–180. ACM, 2007.

[66] Rada Mihalcea. Language independent extractive summarization. In *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, pages 49–52. Association for Computational Linguistics, 2005.

[67] Rada Mihalcea and Hakan Ceylan. Explorations in automatic book summarization. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 380–389, 2007.

[68] Peter Mika. Ontologies are us: A unified model of social networks and semantics. In *The Semantic Web–ISWC 2005*, pages 522–536. Springer, 2005.

[69] Gilad Mishne. Autotag: a collaborative approach to automated tag assignment for weblog posts. In *Proceedings of the 15th international conference on World Wide Web*, pages 953–954. ACM, 2006.

[70] Gilad Mishne, David Carmel, and Ronny Lempel. Blocking blog spam with language model disagreement. In *AIRWeb'05: Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web*, pages 1–6, 2005.

[71] Gilad Mishne and Natalie Glance. Leave a reply: An analysis of weblog comments. In *Third annual workshop on the Weblogging ecosystem*, 2006.

[72] Mor Naaman, Jeffrey Boase, and Chih-Hui Lai. Is it really about me?: message content in social awareness streams. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 189–192. ACM, 2010.

[73] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[74] D. Radev, S. Teufel, H. Saggion, W. Lam, J. Blitzer, A. Celebi, H. Qi, E. Drabek, and D. Liu. Evaluation of text summarization in a cross-lingual

information retrieval framework. *Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD, Tech. Rep*, 2002.

[75] D.R. Radev, E. Hovy, and K. McKeown. Introduction to the special issue on summarization. *Computational linguistics*, 28(4):399–408, 2002.

[76] Dragomir R Radev, Sasha Blair-Goldensohn, and Zhu Zhang. Experiments in single and multi-document summarization using mead. *Ann Arbor*, 1001:48109, 2001.

[77] Adam Rae, Börkur Sigurbjörnsson, and Roelof van Zwol. Improving tag recommendation using social networks. In *Adaptivity, Personalization and Fusion of Heterogeneous Information*, pages 92–99, 2010.

[78] Daniel Ramage, Paul Heymann, Christopher D Manning, and Hector Garcia-Molina. Clustering the tagged web. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 54–63. ACM, 2009.

[79] Reuters. Usa today. youtube serves up 100 million videos a day online, July 2006.

[80] Daniel M Romero, Brendan Meeder, and Jon Kleinberg. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 695–704. ACM, 2011.

[81] G. Salton, A. Singhal, M. Mitra, and C. Buckley. Automatic text structuring and summarization. *Information Processing & Management*, 33(2):193–207, 1997.

[82] D Sculley and Gabriel M Wachman. Relaxed online svms for spam filtering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 415–422. ACM, 2007.

[83] Shilad Sen, F Maxwell Harper, Adam LaPitz, and John Riedl. The quest for quality tags. In *Proceedings of the 2007 international ACM conference on Supporting group work*, pages 361–370. ACM, 2007.

[84] Chirag Shah. Tubekit: a query-based youtube crawling toolkit. In *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*, pages 433–433. ACM, 2008.

[85] Börkur Sigurbjörnsson and Roelof Van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th international conference on World Wide Web*, pages 327–336. ACM, 2008.

[86] Yang Song, Ziming Zhuang, Huajing Li, Qiankun Zhao, Jia Li, Wang-Chien Lee, and C Lee Giles. Real-time automatic tag recommendation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 515–522. ACM, 2008.

[87] Sanjay Sood, Sara Owsley, Kristian Hammond, and Larry Birnbaum. Tagassist: Automatic tag suggestion for blog posts. In *Proceedings of the international conference on weblogs and social media (ICWSM 2007)*, 2007.

[88] Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 841–842. ACM, 2010.

[89] Gabor Szabo and Bernardo A Huberman. Predicting the popularity of online content. *Communications of the ACM*, 53(8):80–88, 2010.

[90] Jaime Teevan, Daniel Ramage, and Merredith Ringel Morris. # twittersearch: a comparison of microblog search and web search. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 35–44. ACM, 2011.

[91] I. Titov and R. McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM, 2008.

[92] Ivan Titov and Ryan McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM, 2008.

[93] Jian Wang, Liangjie Hong, and Brian D Davison. Rsdc'09: Tag recommendation using keywords and association rules. *ECML PKDD discovery challenge*, pages 261–274, 2009.

[94] Zhonghui Wang and Zhihong Deng. Tag recommendation based on bayesian principle. In *Advanced Data Mining and Applications*, pages 191–201. Springer, 2010.

[95] M.J. Witbrock and V.O. Mittal. Ultra-summarization (poster abstract): a statistical approach to generating highly condensed non-extractive summaries. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 315–316. ACM, 1999.

[96] Zhichen Xu, Yun Fu, Jianchang Mao, and Difu Su. Towards the semantic web: Collaborative tag suggestions. In *Collaborative web tagging workshop at WWW2006, Edinburgh, Scotland*, 2006.

[97] Jaewon Yang and Jure Leskovec. Patterns of temporal variation in online media. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 177–186. ACM, 2011.

[98] Wai Gen Yee, Andrew Yates, Shizhu Liu, and Ophir Frieder. Are web user comments useful for search. *Proc. LSDS-IR*, pages 63–70, 2009.

[99] Zhongwu Zhai, Bing Liu, Hua Xu, and Peifa Jia. constrained lda for grouping product features in opinion mining. pages 448–459. Springer, 2011.

[100] Zi-Ke Zhang, Tao Zhou, and Yi-Cheng Zhang. Personalized recommendation via integrated diffusion on user–item–tag tripartite graphs. *Physica A: Statistical Mechanics and its Applications*, 389(1):179–186, 2010.